Parallel simulation of hydraulic systems using transmission-line modelling (TLM)

Burton, James D.

Award date:
1994

Awarding institution:
University of Bath

PARALLEL SIMULATION OF HYDRAULIC SYSTEMS USING TRANSMISSION-LINE MODELLING (TLM)

Submitted by James D Burton

for the degree of PhD

of the University of Bath

1994

James D Burton

ProQuest LLC
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106-1346

SUMMARY

Computer simulation is a powerful tool for the design, analysis and synthesis of hydraulic systems. A simulation of system dynamic performance could be used as a basis for automated condition monitoring, a model reference for predictive control, or to tune the operation of existing plant to improve efficiency.

Components in a computer simulation of a hydraulic system are often represented by ordinary differential equations (ODE's) and solved in the time-domain using conventional numerical integration techniques. However, the solution of the wave equation in fluids gives rise to a form of mathematical description known as transmission line modelling (TLM). Using the TLM approach fluid volumes can be modelled by the propagation of pressure and flow waves at the speed of sound in the hydraulic fluid. The finite transmission delay due to the limited rate of information propagation across a fluid volume decouples the components at either end of the volume. It is therefore possible to solve each component model independently, with an exchange of information at the end of each solution step.

A significant advantage of allowing component models to be solved separately is that each component model, or sub-circuits of component models, can be solved on separate "parallel" computers. Each component model is entirely self-contained and numerically isolated from components connected to it by finite transmission delays. In partitioning components onto separate processors therefore, both the models that represent the circuit and the mathematical operations required to compute the solution are fully distributed.

The research described in this thesis examines the use of TLM to decouple the solution of hydraulic system component models, and moves on to describe the implementation of a TLM simulation package, for the general prediction of hydraulic system performance. It has been demonstrated that considerable improvements in simulation speed are possible using the TLM method on a single processor, and that significant additional gains are possible by partitioning a TLM simulation onto two or more processors.

Depending upon the numerical nature of the equivalent ODE simulation, the speed increase achieved employing TLM ranged by an order of magnitude from approximately three to thirty times faster. The algorithmic "speed-up" afforded by the TLM solution technique was found to be highly problem

dependent.

The uneven distribution of processor computational loads, in addition to frequent communications between sub-circuit "partitions" executing on different processors, was found to have a very detrimental effect on the performance of the parallel TLM solver. A two or three fold increase in execution speed compared to a single processor, using four processors, was the maximum attainable for the simulation examples considered.

FOREWORD

During the course of this study a total of four papers were written and presented by the author jointly with Professors Kevin A Edge and Clifford R Burrows. Copies of these papers have been appended to the back of this thesis for further information, and in some cases to avoid unnecessary repetition of earlier work.

ACKNOWLEDGEMENTS

Firstly, I would like to express my gratitude to my supervisors Professors Kevin Edge and Clifford Burrows, for their invaluable help and advice, and to the Science and Engineering Research Council for funding the three year research program that made this work possible.

My sincere thanks are also due to colleagues at the Fluid Power Centre, both past and present, many of whom have made this a most memorable and rewarding part of my career.

Finally, I owe my biggest thanks of all to my wife Vanessa, for her understanding, continued support and encouragement.

LIST OF CONTENTS

LIST OF TABLES AND FIGURES

NOMENCLATURE

A	Area
a	Acceleration
B	Bulk Modulus
C	Characteristic Pressure, Coefficient (orifice model)
c	Speed of sound
d	Pipe diameter
D	Fluid displacement (of pump or motor)
F	Force
G	Shear modulus
g()	Denotes any general function
h()	Denotes any general function
I	Fluid inertance
J	Mechanical inertia
K	Flow-pressure gradient, Orifice constant, Spring rate (use described in text)
L	Length
M	Mass
N()	Loss function used in 2-port, 4-pole transmission line
n	Gear reduction ratio
P	Pressure
Q	Flow
R	Laminar friction coefficient
r	Pipe radius
s	Laplace operator, Actuator stroke (use described in text)
T	Transmission delay
t	Time
u	Time-varying input demand
V	Volume
v	Velocity

x	Linear displacement
Z	Characteristic surge impedance for fluid-filled pipeline
α	Filter coefficient used in capacitive line model
Δ	Simulation time step
γ	Ratio of new to old line impedance
ΔP	Differential pressure
γ	Impedance ratio
λ	Convolution time
μ	Dynamic viscosity
ν	Kinematic viscosity
ρ	Fluid density
τ	Time constant
ϕ	Distributed laminar friction coefficient
χ	Frequency-dependent friction dispersion coefficient
ψ	Frequency-dependent friction dispersion coefficient

Subscripts

a	Line end 'a', Accumulator (in accumulator model)
b	Line end 'b'
c	Cracking (relief/check valve models), Clearance (pump model)
cav	Cavity volume (cavitation)
ch	Charged (accumulator model)
cyl	Cylinder (accumulator model)
d	Discharge (coefficient in orifice model)
e	Effective value (bulk modulus), Error (time step controller), End (cavitation model)
f	Coulomb friction (actuator and motor models)
g	Gas (accumulator model)
i	Integer index
J	Junction (to connect multiple component models)

L	Laminar (laminar resistor model), Externally applied load (actuator/motor models), Transmission line (cavitation)
m	Motor (hydraulic motor model), Component model (cavitation)
n	Integer step number
o	Orifice
p	Pump (hydraulic pump model)
pm	Prime mover
R	Resistor (laminar resistor model), Resistive (torque/force in motor/actuator model)
ref	Reference value
S	Stiction (force/torque)
s	Start (cavitation), Slip (loss in pump/motor model), Shaft
t	Tank (reservoir model)
tol	Tolerance (used in stick-slip friction modelling)
v	Vapour (cavitation), Volume, Valve (valve models), Viscous (loss in motor model)
ω	Denotes a torsional system

Superscripts

a	Step controller exponent (accepted step)
b	Step controller exponent (accepted step)
c	Step controller exponent (step rejection)
n	Gas polytropic index (accumulator model)
^	Denotes a maximum value
'	Denotes a new value

[Note: some notation is described fully in the main body of the thesis where it appears and is not referred to in the above list]

CHAPTER 1

INTRODUCTION

§1.1 Requirement

Large and complex hydraulic systems are employed in a diverse range of high risk, high capital investment applications, such as in the aerospace, marine and mobile equipment industries. Consequently, safety, integrity and efficient operation are of critical importance. Computer simulation is now used extensively in the analysis, synthesis and design of such systems, because the complex interactions that may occur between components are difficult, if not impossible to examine otherwise. The speed of a simulation is governed by the detail with which a component is modelled and the number of individual components in a system. In order to analyze larger and more complex systems within an economic time frame, faster methods of computation are required.

Typically simulation takes a serial approach to computing a solution, each numerical calculation is performed in sequence by a single processor. Serial processing, even using a very expensive super-computing platform, is limited to a theoretical maximum speed, dictated by the design of the central processing unit (CPU). A faster simulation can be obtained if two, or more numerical calculations are performed by different processors simultaneously.

Parallel computation using a number of separate, communicating processors represents a viable and cost effective alternative to computers that employ a single but extremely fast processor. "Supercomputers" can use several processors in parallel, using techniques such as pipelining to schedule tasks to improve performance [Hockney & Jesshope, 1981]. This approach is not always appropriate in engineering because many of the applications that would benefit from faster processing cannot justify the cost (or physical size) of a supercomputer.

Sufficient gains in execution speed will enable real-time hydraulic system modelling, which has

significant implications in the areas of non-linear model reference adaptive control (MRAC), sophisticated on-line condition monitoring and for examining the interaction of simulated plant with real hardware, or human operators.

The majority of MRAC techniques so far developed rely on a simple discrete transfer function representation of the system model, but significant limitations exist due to unmodelled dynamics [Craig, 1988]. The increased performance offered by parallel processing will allow more complex representation of the system model, incorporating highly non-linear effects such as flow through a turbulent orifice, which would otherwise be linearised.

The potential benefits of on-line condition monitoring to engineering systems are considerable; critical operating conditions can be detected in advance and automatic corrections, or warnings, initiated. Conventional techniques involve monitoring parameters such as leakage rate, pressure ripple, temperature and vibration signatures over long periods of time for nominally constant operating conditions [Hunt, 1986]. Abnormal operation is detected only when a measured parameter moves outside a specified window of operation. If parallel processing is used to provide an advanced non-linear plant model at sufficient speed, then it will become possible to identify parameters such as damping coefficient, friction, or leakage rate from the dynamic performance of the system in real time [Sato *et al*, 1991]. Such parameters can be determined by forcing the plant model to track the real plant dynamics using a minimum of plant states, given significant but easily obtainable *a priori* information about the real plant such as physical dimensions, mass, spring rates etc. Examples of hydraulic plant condition monitoring using this technique include estimates of bearing wear from friction, or seal condition from leakage rate.

Parameter identification using dynamic information from the plant can enable incipient faults to be detected in advance of conventional methods that rely on steady-state measurements [Isermann, 1991]. Moreover, for a hydraulic system that undergoes a complex operating cycle, conventional condition monitoring techniques are inappropriate and the direct comparison of the plant dynamic performance with an advanced real-time dynamic model is likely to be the most reliable and effective solution to the problem of unmodelled plant dynamics.

Another area of growing interest is that of multi-variable plant optimisation [Krus *et al*, 1991A]. Increasingly, genetic algorithms are employed in order to detect the existence of genuine minima, or maxima (optimum values, given specified performance criteria) with respect to important plant

variables, such as physical component sizes and control settings. This necessitates many (typically, in excess of one thousand) dynamic simulations of the same plant model, with parametric variations (eg. pump displacement, controller gain etc.). For large, complex hydraulic circuits the computer run times required for a single simulation implies that optimisation is likely to be prohibitive. Performance increases, obtained by restructuring the plant model into parallel processes, will improve the viability of the optimisation process, notwithstanding the significant gains already obtained by performing parametric variants of sequential simulations on separate processors [Donne *et al*, 1993]. In future, the use of a "massively parallel" machine could enable many different parallel simulations to be executed simultaneously.

A material increase in performance will enhance the iterative design process, where dynamic simulation forms an integral part of that process in order to reduce the number of costly prototype tests. It is frequently very time consuming for the hydraulic system designer to investigate the transient response of many different hydraulic systems using computer simulation.

The work described in this thesis is aimed at developing a parallel simulation methodology to facilitate both the rapid and accurate dynamic analysis of large and complex hydraulic systems. An important part of this research addresses the problem of the structure and design of such a system, in conjunction with the mechanism for decoupling and hence partitioning the numerical simulation into parallel tasks. The application of these techniques to example systems and investigating performance for alternative parallel configurations are significant aspects of research contribution.

§1.2 Background

Most conventional serial process simulations are assembled from lumped-parameter component models, using ordinary differential equations (ODE's) and algebraic equations to represent the physical circuit elements [Tomlinson, 1987]. Newton's second law of motion, fluid compressibility and turbulent flow through an orifice are typical of the phenomena that are modelled using ODE's or algebraic lumped-parameter equations. Numerical stiffness and frequent discontinuities are common to lumped-parameter fluid power system simulation.

A *stiff* system is one where the dynamic behaviour is described by a set of coupled differential equations, which have solutions with widely differing decay rates. The smallest time constant (largest eigenvalue), however, dictates the maximum time step allowable for numerical stability [Lambert,

1973] [Hairer & Wanner, 1991]. It is this characteristic of stiff systems that results in very small time steps and correspondingly long computer run times. Irritatingly, it is the physical parameters associated with this time constant that decay most rapidly and contribute least to the salient dynamics.

A serious problem arises when the equations describing system behaviour are discontinuous in nature, because of the occurrence of a physical discontinuity. A discontinuity is characterised by a step change in a model parameter, which can cause instability in the numerical integration process unless the algorithm is capable of detecting the discontinuity. Physical discontinuities may be divided into two categories; a change in input demand at known times, or a model parameter reaching a critical value, eg. when an actuator reaches its travel limits, or when a valve opens, resulting in an instantaneous change.

The current state-of-the-art in the field of numerical integration is the variable method (type-insensitive), variable order and variable time step integration algorithm, eg. LSODA [1] [Petzold, 1983] [Richards *et al*, 1990]. Such algorithms have been designed specifically for use with stiff systems of differential equations.

The parallelisation of a system may take the simple approach of partitioning the current lumped parameter models and integration algorithms to run on separate processors. This is termed functional parallelisation, because the problem is decomposed into clearly identified functional blocks. The limitations of this approach are that the partitioning may not be evenly balanced, resulting in one processor performing the majority of the computational load. An alternative approach is algorithmic partitioning, in which the parameter models and integration algorithm are written in a form more suited to parallel computation. It is this algorithmic partitioning which is used in the transmission line modelling approach (TLM).

TLM utilises the solution of the loss-less wave equations, thereby allowing for the transport delay that occurs in transmission systems. The transport delay (which is the time required for a wave to propagate along a transmission line) effectively decouples one end of the line from the other. This is a distributed parameter model (lumped parameter models are actually derived as approximations to distributed parameter models, where the transport delay is assumed to be negligible).

[1] LSoda uses either GEAR, or ADAMS integrators, depending upon system stiffness.

Currently, there exists little research concerned directly with the application of parallel computing to the dynamic simulation of fluid power systems. Krus *et al* [1990] demonstrated the use of two personal computers linked via a serial data cable to perform the parallel simulation of a simple hydraulic circuit example, using a fixed time step transmission-line modelling (TLM) algorithm. More recently, Jansson *et al* [1992] described the use of a variable time step TLM algorithm, using different time steps in different sub-system partitions. This approach was evaluated by employing different processes for each sub-system on the same processor, communicating via shared memory. The effectiveness of this procedure is very dependent upon the partitioning arrangement chosen, because of the different time steps required by different partitions; some parts of the system may be isolated artificially by the partitioning and critical interactions between partitions lost. To avoid such difficulties relies to a large extent on significant *a priori* knowledge of the system dynamics. Moreover, some very disturbing time step size oscillations were introduced, as a consequence of synchronizing the partitioned processes.

A similar approach has been employed by other researches using conventional integration methods. Harris [1990] used Gear's method to solve the hydraulic part of system, whilst using simple Euler integration to solve for thermal transient effects. This has proven to be an acceptable approach because the coupling between the hydraulic and thermal parts of the system is weak; fast hydraulic transients have little or no affect on the thermal transient behaviour of the system.

A stiff integrator could be applied to separate partitions in the hydraulic simulation if the coupling between the partitions is relatively weak. If this were not the case then the error control algorithm within each partition would interact with the other, forcing the use of small time steps. The nature of the communications is also problematic if variable time steps are used; constraining the integrators to synchronise may cause time step oscillations as in Jansson's TLM algorithm; interpolation will introduce additional errors.

§1.3 Scope

This thesis is aimed at a more in-depth evaluation of the TLM approach to the parallel simulation of hydraulic systems. The current availability of low cost hardware platforms and the recent development of software for parallel programming allows TLM to be developed in a true multi-processor environment. However, this technology is constantly improving in terms of ever-increasing

processor speed and as such this research is concerned with the relative performance improvements possible using parallel processors, rather than any absolute measure of performance.

It is particularly important to compare the speed advantages of a multi-processor simulation against the use of a single processor of the same type. A feature of parallel processing is that the partitioning of a simulation onto several processors introduces a communications overhead, which can actually reduce the speed of simulation below that of a single processor.

The concepts of processor load balancing and the communications penalty between processors are presented and evaluated for specific test system simulations, and partitioning guidelines for efficient parallel operation are reported. Larger and more complex systems, incorporating a diverse range of component models, are investigated using TLM. Detailed models of complex systems are partitioned into a number of sub-systems, placed on separate processors. Furthermore, a numerical scheme using instantaneous signal links is devised for hydraulic systems that involve electro-hydraulic control, whilst retaining the TLM representation of the hydraulic lines to facilitate the decoupling of hydraulic component models. The constraints imposed when configuring a parallel implementation of such systems are addressed and solutions evaluated.

§1.4 Organisation of Thesis

The remainder of this thesis is divided into a further six chapters, structured in the following way:

Chapter 2. Chapter two reviews the possible alternatives for the parallel, dynamic simulation of hydraulic systems and establishes the essential criteria for an effective and efficient numerical approach to the parallelisation problem. Reasons for adopting the TLM-based approach are discussed.

Chapter 3. Chapter three details the modelling of hydraulic lines using TLM, with reference to the use of error control and time step size selection for efficient operation and for the treatment of numerical discontinuities. The modelling of phenomena such as line friction, cavitation, variable volume and variable bulk modulus using TLM-based algorithms is also discussed, in addition to the mechanism employed for instantaneous control signals used in electro-hydraulic sub-circuits.

Chapter 4. This chapter describes the design of the TLM component models, central to the formation of TLM hydraulic system simulations. Specific component model examples, used in the system simulations of Chapters five and six, are detailed.

Chapter 5. Here, the structure and methodology devised for reconfigurable, component model based simulations using a *program generator* is reported and the implications for parallel operation outlined. A specific example system is simulated, the "two-actuator" circuit, using a multi-step TLM algorithm. Explicit comparison with the lumped parameter equivalent simulation is undertaken and evaluated in terms of the correspondence of results and computational performance on a single processor.

Chapter 6. Chapter six evaluates the parallel TLM implementation, with reference to processor configuration and the nature and topology of the hydraulic system being simulated. The methodology adopted for circuit partitioning into sub-systems is described in greater detail. Specific example circuits are partitioned and the performance of the parallel simulations quantified for different numbers of processors and for different component model to processor mappings (partitioning arrangements). Guidelines essential to efficient parallel operation are described, to ensure the adequate balance of computational loads and to minimise the overhead associated with data transfer between processors.

Chapter 7. Finally, chapter seven summarises the conclusions drawn from the various aspects of this research, in particular the effectiveness and potential of TLM for parallel simulation. Furthermore, the current status of the system is outlined and some possible areas for future investigation are suggested.

CHAPTER 2

ALTERNATIVES FOR PARALLEL SIMULATION

§2.1 Introduction

Fundamentally important to parallel simulation is the efficient means of distributing the computation onto multiple processors. This chapter evaluates alternative techniques to facilitate the partitioning and hence the parallel simulation of fluid power system dynamics.

§2.2 Lumped Parameter Modelling

Realistic modelling of hydraulic plant frequently necessitates a large number of state variables, if all dynamic elements in the system are to be represented by simultaneous first order differential equations. These systems are often characterized by numerical difficulties, such as strong non-linearities and highly coupled, stiff, differential equations. In addition, the numerical models may incorporate severe discontinuities, with different modes of operation modelled by different continuous equations [Richards *et al*, 1990]. As a consequence the numerical integration algorithms necessary to deal with the stiffness problem have to locate each discontinuity and restart the integrator at this point to avoid a numerical failure. Such numerical procedures usually result in long computer run times. Moreover, severe variations in simulation speed are typical and are attributable to changes in the stiffness of the simulated system as the simulation progresses. Such performance characteristics are not well suited to real-time applications, where the plant model execution speed must be predictable and preferably constant.

As an example consider the hydraulic circuit outlined in Figure 2.1, which shows an application that involves the synchronized operation of two actuators. The simplest lumped-parameter model of this circuit considers each group of inter-connected pipelines as a lumped volume, which accounts for

compressibility effects only. All components attached to the volume (pumps, valves, actuators etc.) calculate flows in response to the same fluid pressure. Component models in the circuit are directly connected by volume pressure and are therefore closely coupled. This coupling between component models is highly dependent upon the line compressibility, or more precisely capacitance (ratio of volume to effective bulk modulus) in the lumped line model; very small volumes result in correspondingly fast pressure transients. The requirements of a stable and efficient numerical integration algorithm for such systems are therefore very demanding, particularly for large systems of state variables. The lumped parameter simulation of the circuit shown in Figure 2.1 (described more fully in §5.4) contains twenty-nine models, most of which include discontinuities and non-linear elements, involving a total of eleven state variables. A simulation time of only two and a half seconds, required approximately 4860 CPU[2] seconds of processing time on a SUN Sparc 4/370; very much slower than real-time. Numerically, this is a highly stiff, non-linear lumped-parameter simulation, which required 561 Jacobian re-evaluations with simulation time steps ranging from 3.16×10^{-16} to 2.6×10^{-2} seconds.

§2.3 Distributed Parameter Modelling

Real-time simulation of the circuit described in Figure 2.1 is only feasible with substantially increased processor speed, or by partitioning the simulation into parallel tasks on individual processors. Parallel operation requires a convenient method for dividing a hydraulic system into separate numerical tasks. Partitioning of the lumped numerical analysis, as mentioned above, is non-trivial and highly system dependent. Links of *weak coupling* (large volume lumped lines, for example) could provide convenient points at which to divide a simulation. However, the Jacobian matrix describing the system dynamics changes significantly with strong non-linearities and at discontinuity points, which in turn may change the nature of the coupling between components. Such behaviour would require dynamic reconfiguration of the parallel implementation to remain efficient, accurate and numerically stable.

A preliminary study of a variety of complex circuits [Burton, 1990] revealed that most processing time is actually spent performing centralised integration, as opposed to solving the model equations

[2] Central Processing Unit

(ie. evaluating rates of change of state variables and passing these to the integrator). In the case of the circuit shown in Figure 2.1, for example, tests revealed that approximately sixty percent of CPU time was used for numerical integration alone.

There is some scope for parallelising the numerical operations within the integrator itself, for example there exist parallel algorithms for L-U factorisations. A type-insensitive integrator such as LSODA [Petzold, 1983] [Richards *et al*, 1990] is well suited to the lumped-parameter simulation of hydraulic systems on a single processor, but is not readily parallelised due to its highly sequential design. Thus only operations that relate to the component models facilitate parallel implementation; the resultant gain in performance would therefore not be significant owing to the computational bottle-neck at the integrator. In practice there will be an additional communications penalty between the partitioned models and the integration algorithm (even if it is to some extent parallelised), which will make matters worse.

A far better approach is to distribute the integration throughout the problem domain into the component models themselves, an option which is offered by Transmission Line Modelling (TLM). This approach utilises the physical transport delay in the pipelines that connect components together. If the transmission of information is restricted to the speed of wave propagation, then there is no immediate communication of information between components connected by *distributed parameter* line models. Consequently, there is no requirement to solve a large, monolithic system of coupled differential equations at every discrete time step, as each component model is decoupled numerically from its neighbouring components.

Some techniques for modelling the effects of distributed parameters do not strictly include a transmission delay, such as in finite element and finite difference methods of solution. The finite element technique, involves transforming the PDE's[3] for fluid momentum and continuity into an arbitrary number of simultaneous ODE's at specified longitudinal intervals, using the Galerkin method for example [Paygude *et al*, 1985]. A lumped-parameter solver, such as LSODA, is then required to obtain a time-domain solution.

The following distributed parameter methods can achieve numerical decoupling, by incorporating a pure transmission delay into the pipeline model.

[3]Partial Differential Equation

§2.3.1 The Method of Characteristics (MOC)

With this method each line is discretised into a series of internal points at the intersection of *characteristic lines*; pressure and flow velocity is then calculated at each of these internal points [Fox, 1977]. Skarbek-Wazynski [1981] investigated the application of this approach to general fluid power systems analysis, but found the method of characteristics unable to model small trapped volumes (eg. those associated with manifolds and fluid volumes in valves etc.) as accurately as the lumped-parameter, compressibility-only model. A combination of MOC and lumped-parameter simulation, with a central integrator to compute small volumes and administer boundary conditions to MOC lines, was determined the most appropriate solution.

§2.3.2 Transmission-line Modelling (TLM)

In this case each line is represented as a four-port transmission-line element. The pressures and flows are only evaluated at each end of the line [Auslander, 1968] [Karam, 1972] [Sidell & Wormley, 1977] [Boucher & Kitsios, 1986] [Krus *et al* 1990]. Transmission-line methods have been applied to electrical, thermal and mechanical, as well as fluid power systems [Johns & O'Brien 1980] [Pulko & Olashore, 1989] [Partridge *et al*, 1987]. Consequently, plant containing all these elements, may be modelled using similar numerical techniques for the hydraulic, mechanical and electrical parts. Figure 2.2 illustrates the example system; this time, however, the pipes are no longer combined into lumped volumes (pure capacitances), but treated as individual transmission-line delay elements, connected by line junctions and components. Computer models now only exist for each component in the system (the line junctions are also types of component).

In most cases, TLM requires less computational effort than the method of characteristics, as no *internal* points are calculated. Moreover, the number of component models forming a system is reduced, as the equations relating pressure and flow at the transmission line *ends* are incorporated into the model. The total number of models is then reduced, as the pipe models are not solved separately, but incorporated into the component model ports. When using MOC each line model must be solved first, using the boundary conditions provided by the connecting component models. This allows individual components to be decoupled by the line models, but results in less numerical stability than TLM, because the boundary conditions between pipe and component are not solved simultaneously.

§2.4 Closure

TLM is a very powerful approach for domain decomposition and hence parallel operation, as all models are self-contained and are inherently decoupled by a transmission delay. Both the computation of component models and the integration process may therefore be fully distributed. This allows individual components, or component sub-circuits (*topological groups*) to be assigned to separate processors and restructured into a partitioned simulation.

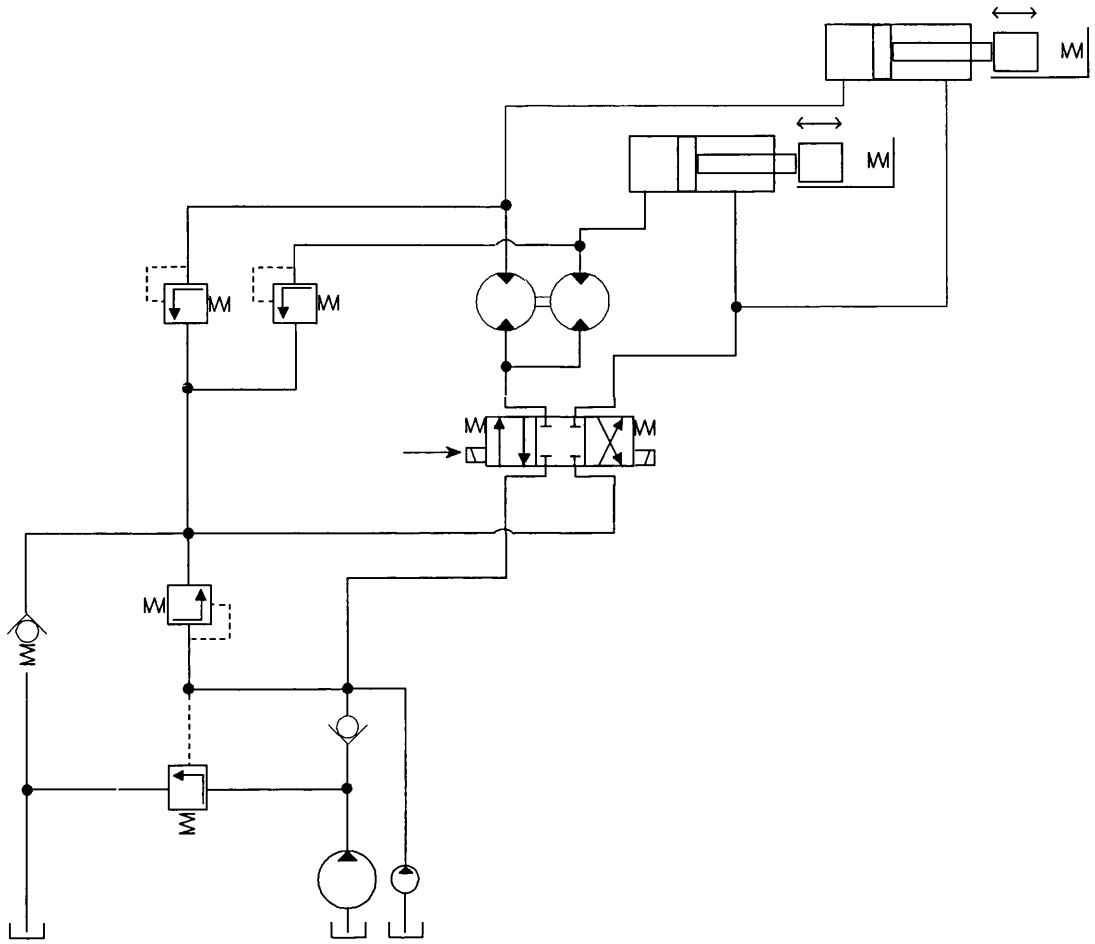


FIGURE 2.1 "TWO-ACTUATOR" EXAMPLE CIRCUIT

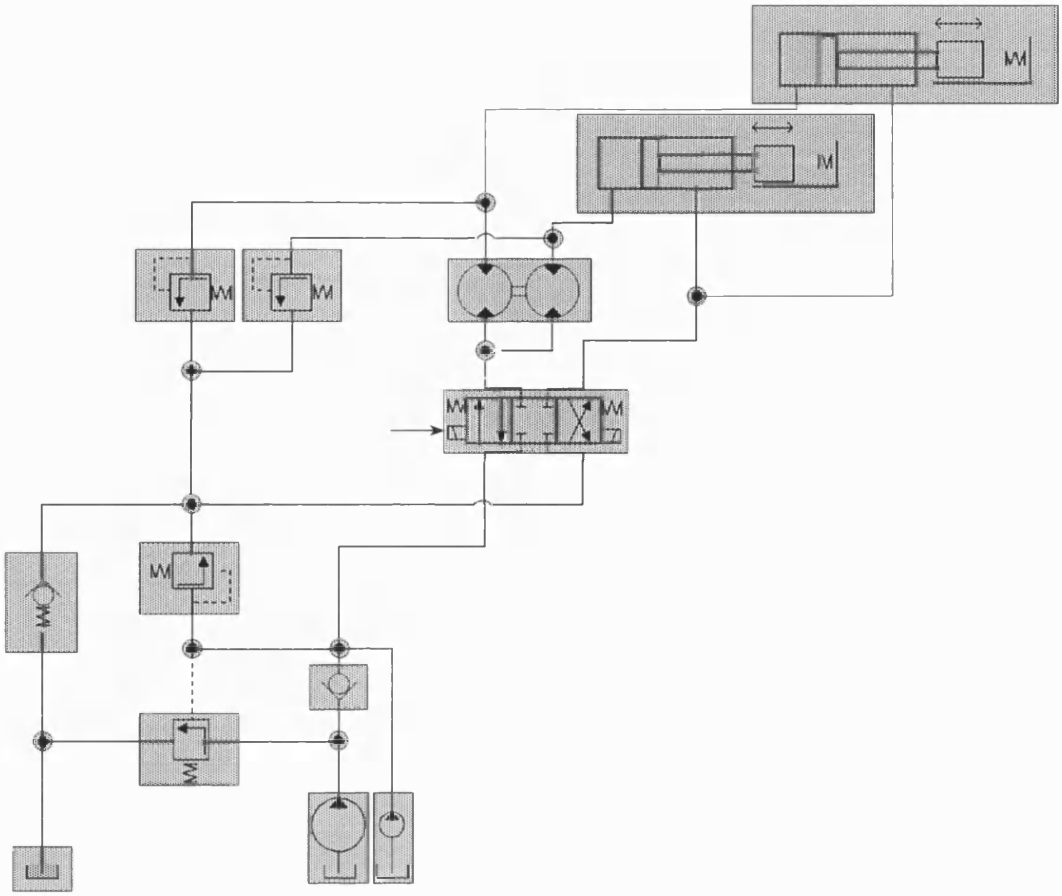


FIGURE 2.2 EQUIVALENT TLM CIRCUIT

CHAPTER 3

TRANSMISSION-LINE MODELLING

§3.1 Introduction

In this chapter computationally efficient transmission-line models for pipelines and fluid volumes are developed, to enable the decoupling of circuit components and facilitate the subsequent partitioning of the problem into parallel tasks. In addition, transmission line models for some fundamental fluid phenomena are outlined, including cavitation, variable fluid volume (eg. displaced liquid in an actuator) and bulk modulus (eg. variation with pressure). Furthermore, the problem of modelling control signals, such as the feedback signal to an electro-hydraulic servo-system, is addressed. This latter problem places constraints on circuit partitioning by introducing an instantaneous (relative to the line transmission delay) connection between component models.

To improve simulation efficiency and to deal with the problem of physical discontinuities, a variable time step transmission-line modelling algorithm is developed. Finally, a line junction model is introduced, which provides a mechanism for connecting multiple components to the same line system, enabling a direct representation of the physical circuit layout.

§3.2 Hydraulic Line Modelling

Time-domain transmission-line equations may be derived from the hyperbolic 2-port 4-pole equations, given by eq.(3.1) [D'souza & Oldenburger, 1964] [Viersma, 1980]. These equations relate to the transmission line shown in Figure 3.1.

$$\begin{aligned} P_a - Z\sqrt{N(s)}\,Q_a &= \left(P_b + Z\sqrt{N(s)}\,Q_b\right) e^{-sT\sqrt{N(s)}} \\ P_b - Z\sqrt{N(s)}\,Q_b &= \left(P_a + Z\sqrt{N(s)}\,Q_a\right) e^{-sT\sqrt{N(s)}} \end{aligned} \qquad (3.1)$$

$$Z = \frac{\rho c}{A}$$

where N is a time-dependent loss coefficient and T is the line propagation delay. The term Z is referred to as the line characteristic surge impedance.

The symmetry of this configuration results in identical forms for the transmission-line equations relating each end of a *loss-less* transmission line, where $N(s)$ in eq.(3.1) is unity [Blackburn, Reethof & Sheerer, 1960]:

$$\begin{aligned}(P_a - ZQ_a)(t) &= (P_b + ZQ_b)(t-T)\\ (P_b - ZQ_b)(t) &= (P_a + ZQ_a)(t-T)\end{aligned} \tag{3.2}$$

These equations describe each end of a distributed line in the time domain. In order to set up a TLM component the line end equation is incorporated into the component model and solved simultaneously. Thus there are no separate line models used in TLM, only component models which have the line models embedded in the form of transmission line ends.

The symmetry of the line equations allows the same sign convention for flow direction at each line end; flows out of the component model (into the line) are defined as positive. This feature enables TLM component models to be connected together to form a system of models in a consistent manner when building a circuit simulation from a standard library of components.

To simplify subsequent analyses it is convenient to introduce a *characteristic pressure*, C, such that:

$$\begin{aligned}C_b &= P_b + ZQ_b\\ C_a &= P_a + ZQ_a\end{aligned} \tag{3.3}$$

This represents the information that must be propagated at a finite transmission speed between transmission line ends, ie. between connected TLM component models. The loss-less form of the transmission-line equations is re-written in terms of characteristic pressures, as follows.

$$\begin{aligned}(P_a - ZQ_a)(t) &= C_b(t-T)\\ (P_b - ZQ_b)(t) &= C_a(t-T)\end{aligned} \tag{3.4}$$

To illustrate the solution of the TLM wave equations in the time domain, consider the simple circuit shown in Figure 3.2. This circuit model consists of a constant flow source connected to a fluid-filled transmission line, which discharges into an infinite, constant pressure reservoir (zero gauge pressure) via a laminar restrictor.

The flow source and the inlet port of the laminar restrictor are connected by characteristic pressure

waves, which propagate through the line at a finite speed. Hence there is a finite transmission delay that decouples both right and left travelling characteristic pressure waves.

The flow source, $\mathbf{Q_s}$, is defined as positive into the transmission line. The corresponding pressure at the source exit (transmission line end) is given by direct substitution of this flow into the transmission line end to obtain the following expression:

$$P_s(t) = C_R(t-T) + ZQ_s(t) \tag{3.5}$$

where $\mathbf{C_R}$ is the characteristic pressure wave propagated from the inlet port of the restrictor calculated at the last time step.

From $\mathbf{Q_s}$ and the pressure $\mathbf{P_s}$ determined from the transmission line end equation, the flow source characteristic pressure wave is calculated from the following equation:

$$C_s(t) = P_s(t) + ZQ_s(t) \tag{3.6}$$

This characteristic pressure is received by the inlet of the laminar restrictor at the next time step. The restrictor calculates flow as a linear function of differential pressure between inlet (transmission line end) and outlet (zero pressure reservoir). Hence for the restrictor:

$$Q_R(t) = R_L\left(P_R(t) - P_T(t)\right) \tag{3.7}$$

where:

$$P_T(t) = 0$$

For the restrictor, flow direction is defined positive from the inlet port (left) to the outlet port (right). For the transmission line end, flow is defined positive <u>into</u> the line, or out of the component, ie. in the opposite direction to the restrictor "inlet" port.

Substituting for restrictor flow into the transmission line end equation and re-arranging:

$$P_R(t) - Z\left(-Q_R(t)\right) = C_L(t-T) \tag{3.8}$$

and in terms of inlet port pressure:

$$P_R(t) = \frac{C_L(t-T)}{1+ZR_L} \tag{3.9}$$

This is an explicit equation for restrictor inlet pressure. The flow $\mathbf{Q_R}$ is then determined by substitution into the equation for the laminar restrictor. The characteristic pressure to be propagated

back to the flow source must be computed as follows, noting the transmission line sign convention for flow.

$$C_R(t) = P_R(t) + Z(-Q_R(t)) \tag{3.10}$$

This characteristic pressure is received by the outlet from the flow source at the next time step. Initial conditions for the characteristic pressures are set to the initial line pressure. Thus at time "zero" the characteristic pressures are the same at each end of the line. (Initial flow conditions could also be specified, although this has not been done for the simulations reported in this thesis).

It is clear that the numerical calculations relating to both the flow source and the laminar restrictor models can be computed simultaneously.

Appendix 1 details the results obtained for a simulation of this TLM example circuit, which is compared with the analytical solution of the same system, modelled using lumped parameters.

§3.2.1 Compressible Line

For the majority of industrial hydraulic applications, compressibility effects dominate the dynamic behaviour of the pipelines and fluid volumes, owing to the relatively close proximity of components in the system [Tomlinson, 1987] [Elleman *et al*, 1993]. The modelling errors introduced by such lumped-parameter compressible fluid volume approximations are usually small when compared with the magnitude of modelling errors due to other component models used in a hydraulic system simulation, for example relief valve models that use a linear flow-pressure characteristic as an approximation to the true valve dynamics [Palmberg, 1991] [Elleman *et al*, 1993].

For a compressible transmission-line approximation to have principally capacitive dynamic properties, the capacitance of the transmission line as defined in the following equation must remain constant.

$$C=\frac{V}{B_e} \tag{3.11}$$

In terms of the transmission-line equations this means that the wave propagation delay, T, can be adjusted to match any desired simulation time step, Δ, provided the line impedance, Z, is correspondingly adjusted to maintain the same line capacitance according to the following equation.

$$
\begin{aligned}
Z &= \frac{\rho c}{A} \\
&= \frac{\rho c L}{V} && \left[A = \frac{V}{L}\right] \\
&= \frac{\rho c^2 \Delta}{V} && [L = c\Delta] \\
&= \frac{\Delta}{V/B_e} && \left[c = \sqrt{\frac{B_e}{\rho}}\right]
\end{aligned}
\tag{3.12}
$$

This line model introduces a modelling error into the fluid inertance term, defined in the following equation, as a result of the change in transmission delay.

$$
I = \frac{\rho L}{A} \tag{3.13}
$$

This is an acceptable approach, provided that inertia effects remain small in comparison with capacitive effects by keeping the line delay within realistic limits; for example a 1m oil-filled line corresponds to a line delay of about 1ms, and so the time step should be around 1ms or less.

Some level of inertance is a realistic effect, as all physical transmission lines, however small, have some associated fluid inertia."Adjusting" each pipe length in a system in order to achieve the same transmission delay, Δ, between components greatly simplifies the numerical procedure, as the same time step may be used to numerically isolate all components connected via transmission lines. Pipeline inertance can be expressed in terms of the simulation time step (or transmission delay) and compressibility as follows.

$$
\begin{aligned}
I &= \frac{\rho L}{A} \\
&= \frac{\rho L^2}{V} && \left[A = \frac{V}{L}\right] \\
&= \frac{\rho c^2 \Delta^2}{V} && [L = c\Delta] \\
&= \frac{\Delta^2}{V/B_e} && \left[c = \sqrt{\frac{B_e}{\rho}}\right]
\end{aligned}
\tag{3.14}
$$

From eq.(3.14) above, the *parasitic* inertia term in the capacitive approximation is proportional to the square of the time step. If the line delay is chosen such that the step size is too large, it is possible that rapid flow transients occurring at either end of the line will be "sampled" at too low

a rate, leading to some distortion of the computed dynamics; a small time step (line delay) is associated with small but frequent characteristic pressure waves, whereas a large time step is associated with less frequent but larger characteristic pressure waves.

To understand the introduction of this distortion due to unwanted inertia effects, it is instructive to consider the transmission line as the sum of a *lumped* compressibility term and a *lumped* inertia term. Re-writing the transmission line equations with lumped parameters (omitting fluid resistance):

$$P_b - P_a + I\frac{dQ}{dt} = 0 \qquad \left[I = \frac{\rho L}{A}\right]$$
$$Q_b - Q_a + C\frac{dP}{dt} = 0 \qquad \left[C = V/B_e\right] \tag{3.15}$$

The additional pressure change due to fluid inertance is the product of the inertia term and the total rate of change of flow into the line. Hence rapid flow transients, coupled with high fluid inertance, results in a correspondingly large and distorted pressure change.

A completely loss-less capacitive transmission line may introduce high frequency resonances into the system model, consistent with the undamped propagation of plane waves. If unrealistic resonances are to be suppressed, then low-pass filtering of the characteristic pressures is necessary and generally approximates the effect of frequency-dependent damping evident in real oil-filled pipelines. The low-pass filtering method described by Krus *et al* [1990] for constant time steps, can also be used with variable time steps [Jansson *et al*, 1992], provided the characteristic pressure waves are compensated as the time step changes (see §3.5). This approach has been used for the subsequent simulation studies discussed in this thesis and is given by the following equation.

$$P_{a,b} - Z' Q_{a,b} = C'_{b,a}$$
$$C'_{b,a}(t) = (1-\alpha)C_{b,a}(t-\Delta_n) + \alpha C'_{b,a}(t-\Delta_n-\Delta_{n-1}) \qquad Z' = \frac{Z}{1-\alpha} \tag{3.16}$$

From the numerical experiments of Krus *et al* [1990] a suggested value for the filter coefficient, α, is 0.2. The use of this recursive algorithm means that the line impedance, **Z**, must be compensated to achieve the same capacitance of the line without filtering.

Characteristic pressure is a combination of both pressure and flow information, which links each end of a fluid-filled transmission line according to a finite delay. Circuit components, such as pumps, valves and actuators, are connected hydraulically by propagating only the characteristic pressure

waves between them. Fluid friction is then most easily represented by the attenuation, or filtering of characteristic pressure waves [Krus *et al*, 1991].

§3.2.2 Line Friction Modelling

The complex nature of line friction requires some simplification when constructing dynamic simulations. A highly complex transmission-line model, involving both steady and unsteady friction components [Trikha, 1975], which may be laminar, or turbulent, is often unwarranted when comparing the relative accuracies of component and line models [Palmberg, 1991]. For most applications frictional losses are dominated by flow controls, such as directional control valves and relief valves. For longer lines where steady-state friction is significant, a lumped friction *orifice* model in series with a compressible transmission-line element is often used [Tomlinson, 1987]. Much longer lines, however, require a more detailed treatment. Essentially, more accurate time-domain solutions are required for eqs.(3.1). For *distributed laminar* friction the loss function, ***N(s)***, is as follows [Krus *et al*, 1991][Viersma, 1980]:

$$N(s) = \frac{R}{ZTs} + 1 \qquad \left[R = \frac{8\mu}{\pi r^4}\right] \tag{3.17}$$

Inverse transformation of the four-pole equations into the time-domain results in the following transmission-line equations, involving a convolution integral at each line end, which are determined by analogy to the electrical transmission-line problem analyzed by Carslaw & Jaeger [1963]

$$P_{a,b} - ZQ_{a,b} = kC_{b,a}(t-T) + \frac{kTAR}{2\rho}\int_{T}^{t} C_{b,a}(t-\lambda)\, e^{\frac{-AR}{2\rho}\lambda}\, \frac{I_1\left(\frac{AR}{2\rho}(\lambda^2-T^2)\right)^{\frac{1}{2}}}{(\lambda^2-T^2)^{\frac{1}{2}}}\, d\lambda \tag{3.18}$$

where

$$k = e^{\left(\frac{ART}{2\rho}\right)}$$

Krus *et al* [1991] suggest that eq.(3.19) is a good approximation to eq.(3.18), which models distributed friction very closely. Bi-linear transformation of this equation (equivalent to trapezoidal integration [D'azzo & Houpis, 1988]) can be used to obtain a discrete time-domain solution, which was the approach used by Krus *et al* [1991]. In their paper very close correspondence with analytical results was reported for the pressure transient at a valve following sudden closure (classical water-

hammer experiment).

$$P_{a,b} - ZQ_{a,b} = \frac{R}{s\phi T + 1} Q_{a,b} + \frac{s\phi T \; e^{-\frac{R}{2Z}}}{s\phi T + 1} C_{b,a}(t-T) \tag{3.19}$$

where ϕ=1.25, which was determined by Krus *et al* [1991] from numerical studies.

Distributed laminar friction gives accurate steady-state losses, but ignores the contribution of unsteady damping (frequency-dependent friction), due to non-plane waves [Brown, 1962] [Trikha, 1975]. The significance of non-plane waves depends upon the magnitude of the *damping number* for the line (otherwise known as the non-dimensional transmission delay), given by the following equation.

$$\tau = \frac{\pi \nu T}{A} \tag{3.20}$$

Goodson & Leonard [1972] suggest heuristics for selecting different line friction models based on damping number (ie. for loss-less, distributed friction, or frequency dependent laminar friction line model).

According to Viersma [1980] the loss function *N(s)* for frequency-dependent laminar friction is given by:

$$N(s) = -\frac{I_o\left(\sqrt{\frac{8ZT}{R}}\, s\right)}{I_2\left(\sqrt{\frac{8ZT}{R}}\, s\right)} \tag{3.21}$$

where $\mathbf{I_0}$ and $\mathbf{I_2}$ are the modified Bessel functions of the zero'th and second kind respectively.

The following approximate loss function is also suggested by Viersma:

$$N(s) \approx \frac{R}{ZTs} + 1 + \sum_{i=1}^{3} \frac{K_i}{\tau_i s + 1}$$

$$K_1 = \frac{0.5}{3.3} \qquad K_2 = \frac{4.05}{25} \qquad K_3 = \frac{20}{1000} \tag{3.22}$$

$$\tau_1 = \frac{ZT}{3.3R} \qquad \tau_2 = \frac{ZT}{25R} \qquad \tau_3 = \frac{ZT}{1000R}$$

Further time-domain solutions to eq.(3.1) are possible, resulting in (computationally-expensive) numerical convolution integrals that must be solved at each time step and at each transmission line

end. Sidell & Wormley [1977] used this technique for predicting the fluid transients in *lumped-distributed* pneumatic networks. If a truncation error in the convolution weighting function is specified such that small terms in the convolution integral are omitted, faster results may be obtained at the expense of accuracy.

Alternatively, simple linear transfer function approximations can be derived for the line impulse response, which is the weighting function used in the convolution integral. Reformulation of the convolution into a recursive algorithm results in computationally fast long-line models. Karam & Leonard [1973], Karam & Tindall [1975] and recently Krus *et al* [1991] have developed such *fast* pipeline models, but rely on a single first order lag term to model the unsteady friction component (wave dispersion). Higher order approximations are possible using additional terms with improved accuracy, although there is a compromise between increased accuracy and increased complexity. Burton *et al* [1992] achieved good results using three first order terms to approximate the unsteady friction component, as shown in eq.(3.23).

$$P_{a,b} - ZQ_{a,b} = \frac{R}{s\phi T + 1} Q_{a,b} + \sum_{i=1}^{3} \left(\frac{\chi_i \psi_i}{s + \psi_i} \right) \frac{s\phi T \, e^{-\frac{R}{2Z_o}}}{s\phi T + 1} C_{b,a}(t - T) \qquad (3.23)$$

However, such approximations strictly apply for laminar flow conditions. Long-line models involving turbulent flows are considerably more complex mathematically, when using the four-port transmission-line representation. A more flexible approach to the above is to sub-divide the line into discrete elements [Viersma, 1990], whereupon more complex friction models may be incorporated. This also allows longitudinal variations in geometry, effective bulk modulus etc to be taken into account.

§3.3 Cavitation Modelling

Detailed modelling of cavitation is highly complex, involving the prediction of liberated air, and saturated vapour, which may be distributed throughout the line as bubbles, or concentrated in a cavity, or a combination of both. A straightforward approach, capturing the most significant aspects of the transient behaviour, is required if the effects of "cavitation" are to be are to be included in all component models. Of all the various cavitation models, vaporous column separation is the simplest to accomplish.

Vapour-only cavitation models generally over-predict the most damaging pressure transients [Karam, 1974] and are attractive because of their inherent simplicity. Burton *et al* [1992] demonstrated the application of such a void tracking algorithm by comparison with the experimental results obtained by Kojima & Shinada [1984]. (To obtain good correlation between numerical calculation and experimental measurements it was necessary to employ the frequency-dependent laminar friction model, described in the previous section).

If the quantity of released air in the void is known, or calculable, the partial pressure of the air may be incorporated into the pressure boundary condition, improving accuracy. This has not been incorporated into the modelling algorithms so far developed. It is also possible to enhance the model further by incorporating delayed air release [Karam 1974], although the added complexity and uncertainty regarding the additional parameters limit this model's usefulness and generality.

Tomlinson [1987] models cavitation by modification of the effective bulk modulus to account for the release of dissolved air according to Henry's Law. The air is assumed to come out of solution as easily compressed pockets of gas the instant the line pressure is reduced below ambient pressure (ie. the pressure at which air is initially dissolved into the hydraulic fluid). This procedure can be adopted by TLM by incorporating transmission lines with variable bulk modulus (capacitance), as described in §3.4. The validation of this and other models against experimental data is an ongoing topic of research.

In this work, a void tracking algorithm is implemented by interposing a cavity between component and line, illustrated diagrammatically in Figure 3.3. When the local pressure falls below the saturated vapour pressure the transmission line boundary condition is changed. For example, if the component model normally provides a flow, Q_m, to solve pressure from the transmission-line end equation, then if $P_m<P_v$, P_v becomes the new boundary condition to the transmission-line end. The flow into the transmission-line, Q_L, is now calculated according to the saturated vapour pressure P_v. The difference between the line flow and the component flow is integrated to obtain a cavity volume, V_{cav}. Only when the cavity volume is restored to zero will cavitation cease. The full equation set for the vaporous cavitation algorithm is as follows.

No cavitation:

$$V_c = 0 \text{ and } P_L \geq P_v:$$

$$P_m = P_L \qquad (3.24a)$$

$$P_L = C_L(t-\Delta) - ZQ_L$$

Cavitation:

$$V_{cav}>0 \text{ or } P_L<P_v:$$

$$P_m = P_v$$

$$P_L = C_L(t-\Delta) - ZQ_L \qquad (3.24b)$$

$$V_{cav} = \int_{t_s}^{t_e} (Q_L - Q_m)dt$$

[Note: in the implementation of the algorithm the cavity volume is prevented from becoming negative by setting it to zero when the cavity collapses.]

§3.4 Variable Line Volume and Bulk Modulus

Many hydraulic components, such as actuators and accumulators, will change the working volume of fluid in a transmission-line (these variable component volumes can be added to the constant line volume). In addition, variations in effective bulk modulus with pressure, for example, will have an effect on system performance; the effective bulk modulus is derived from the isentropic tangent fluid bulk modulus and adjusted to account for the pipe wall stiffness. (Note that if the fluid bulk modulus is assumed constant, the isentropic secant bulk modulus is used).

Using TLM the basic approach to account for changes in volume/bulk modulus is to modify the line characteristic impedance, Z, according to eq.(3.12), as the line capacitance changes, where Z' is the new line impedance. Since the capacitance is varied as the pressure waves propagate along the transmission line, these characteristic pressures must be adjusted to maintain "total pressure" and "total flow" in the line [Jansson *et al*, 1992] [Pulko *et al*, 1990]. The total pressure condition is satisfied if the sum of the characteristic pressures at each line end is the same both before and after the change in line impedance from Z to Z'. For total pressure:

$$C_a + C_b = C_a' + C_b' \qquad (3.25)$$

Total flow through the line is conserved if the difference in characteristic pressures divided by line impedance remains unchanged. For total flow:

$$\frac{C_a - C_b}{Z} = \frac{C_a' - C_b'}{Z'} \qquad (3.26)$$

To meet both constraints the "compensated" characteristic pressures become:

$$C_a' = C_a\left(\frac{1+\gamma}{2}\right) + C_b\left(\frac{1-\gamma}{2}\right)$$

$$C_b' = C_b\left(\frac{1+\gamma}{2}\right) + C_a\left(\frac{1-\gamma}{2}\right) \qquad \left[\gamma = \frac{Z'}{Z}\right] \tag{3.27}$$

§3.5 Variable Time Step Modelling

Fluid inertance is the only mechanism that can result in a transient pressure differential between each end of a "loss-less" transmission-line; filtering of the characteristic pressures according to eq.(3.16) introduces frictional damping to reduce the magnitude of the pressure error, but gives no steady-state pressure loss. Re-formulation into a lumped line model, given by eq.(3.15), shows clearly that any flow transient introduced into the transmission-line results in a corresponding transient pressure difference. Although fluid inertia effects are present in the physical system, an excessive time step may exaggerate this effect greatly, owing to the square-law relationship between fluid inertance and time step given by eq.(3.14).

For a "loss-less" line, the pressure difference can be treated as a pressure error, which indicates how much the line behaviour has diverged from that of an ideal compressible volume. The degree of divergence may be controlled by reconfiguring the TLM solver to use variable time steps. Adjusting the time step and hence the line delay during the simulation, whilst maintaining capacitance constant, implies that the line impedance, **Z**, must be adjusted in accordance with eq.(3.12). In common with modifications to line impedance caused by volume changes, or bulk modulus changes, the characteristic pressure pulses must be adjusted by applying eqs.(3.18). With the aim of reducing simulation run times, Jansson *et al* [1992] proposed a variable time step TLM solver based on this approach and on the previous work of Krus *et al* [1990] and Pulko *et al* [1990].

A variable time step scheme is often necessary to deal efficiently with the significant non-linearities in fluid power systems. Physical discontinuities are often the most difficult non-linearities to handle, and include valves opening or closing, time-varying input demands and actuator travel limits. When an actuator hits its end stop, for example, the piston velocity and therefore the flows at each actuator port undergo what is usually modelled as an instantaneous change. The effects of such discontinuous and highly non-linear flow transients into a transmission-line can be detected as a pressure "error",

given by eq.(3.28), which is an indication of the magnitude of fluid inertia present in the line.

$$P_e = |P_a - P_b| \tag{3.28}$$

Subsequently, the step size may be reduced for the next time step, in order to maintain the pressure error within a prescribed limit. Alternatively, if the error is too large (an order of magnitude larger than the specified limit, for example), then the completed step may be "rejected" and all of the component models called again with a much smaller time step, as discussed below. The process of step rejection can continue until a specified minimum step length is reached; this procedure is a means of discontinuity detection and location.

The scheme used in this research employs a central error controller to monitor the pressure errors associated with all of the transmission-line elements in the circuit model, as shown in Figure 3.4. The maximum value of all pressure errors is then used to determine the step size. The following step selection algorithm has been used in the subsequent simulation studies. In this scheme there is a choice between two different time step control laws, depending upon the magnitude of the maximum pressure error during the simulation compared with the reference value specified.

For the simulation studies reported in this thesis a value of 0.1 bar was used. This figure was found to give rapid operation of the TLM solver, without distorting fluid inertia to the extent that the TLM simulation gave results unlikely to be representative of the physical hydraulic system dynamics (some level of fluid inertia is realistic). Larger reference errors in excess of 1 bar or so result in faster operation, but can give quite different transient results (§5.4.2).

For "normal" operation of the controller the maximum pressure error must be less than the reference error. The current value of time is then increased by the magnitude of the last time step and all of the model states updated. The following control law is then used to select the next time step, which is similar to conventional integration algorithm step control laws [Hairer & Wanner, 1991] [Jansson *et al*, 1992]:

$$\hat{P}_{e(n)} \le P_e^{ref}: \qquad \Delta_{n+1} = \left(\frac{P_e^{ref}}{\hat{P}_{e(n)}}\right)^a \left(\frac{\hat{P}_{e(n-1)}}{\hat{P}_{e(n)}}\right)^b \Delta_n \tag{3.29}$$

$$[a = 10^{-3} \qquad b = 5\times10^{-3}]$$

This control algorithm allows the time step to change progressively within the limits prescribed by the reference error. The coefficient values stated (a and b) have been derived empirically from

numerical experiments and affect both the rate of change of time step and how smoothly that change takes place.

For pressure errors that are much too large to be accepted (mostly at severe discontinuities), the completed time step is "rejected", ie. the current time and the component model states are not updated. In this case the following control law is used to reduce the time step very rapidly, often to the minimum allowable time step set by the user [Hairer & Wanner, 1991] [Jansson *et al*, 1992]. A minimum value of 1μs has proven to be sufficiently small for the examples investigated in this thesis.

$$\hat{P}_{e(n)} > P_e^{ref}: \qquad \Delta_{n+1} = \left(\frac{P_e^{ref}}{\hat{P}_{e(n)}}\right)^c \Delta_n \qquad (3.30)$$

$$[c = 0.7]$$

Following step rejection the simulation process is re-run with a smaller time step to determine a new error and the cycle repeated. In this manner severe discontinuities, eg. a step change in pump delivery, can be located very precisely, because of the very large pressure errors that occur following the flow discontinuity.

Jansson *et al* [1992] recommend low-pass filtering the pressure errors to help smooth the selection of simulation time; the simple filter used is given by the following equation.

$$P_e' = \frac{P_{e(n)} + P_{e(n-1)}}{2} \qquad (3.31)$$

The above procedure is very flexible (by modification of the exponents used in the algorithm) and has been tuned for the hydraulic systems investigated by the author to give a good trade-off between computational speed and "accuracy", although a detailed study was not conducted. In addition to the exponent values specified, parameters such as reference pressure error and minimum time step have a significant effect on operation of the variable time step computation.

Although only capacitive "loss-less" line models have been used with variable time steps in this study, distributed friction "long-line" models may still be used, provided allowance is made for the resistive component of the pressure error. The transmission delay will not, in general, coincide with past data points due to the variable time steps and interpolation of the characteristic pressures will be necessary.

§3.6 Connecting Multiple Components

A potential difficulty exists when trying to connect more than two components to the same fluid volume. To overcome this difficulty, Krus *et al* [1990] have proposed the following method that isolates all components by one time step. All components connected to the transmission-line *volume* element transmit characteristic pressures to and from each other. At any one of these components, the combined characteristics propagated from the other components are simply averaged out, as shown by eq.(3.32). In addition, the impedance of the transmission-line volume is set equal to that given by eq.(3.33), noting that the divisor **1-α** is used to accommodate the gain of the low-pass characteristic-pressure filter [eq.(3.16)].

$$C_i' = \frac{1}{n-1}\left(\sum_{j=1}^{n} C_j - C_i\right) \tag{3.32}$$

$$Z_v = \frac{1}{(n-1)(1-\alpha)}\left(\frac{\Delta}{V/B_e}\right) \tag{3.33}$$

A different approach is adopted here. The connection between several components is handled by the use of a junction, or node model, as depicted in Figure 3.5. The junction is therefore treated in the same way as any other component model in the system, such as a valve, actuator, or pump. The component equations must therefore be solved simultaneously with the transmission-line end equations at each fluid port of the junction, as in the following three-port example.

Junction equations

$$\sum_{i=1}^{3} Q_{ia} = 0 \qquad P_J = P_{1a} = P_{2a} = P_{3a} \tag{3.34}$$

Transmission-line end equations (*i=1, 2, 3*)

$$P_{ia} - Z_i Q_{ia} = C_{ib}(t-\Delta) \tag{3.35}$$

Solving for the junction pressure by substituting the transmission line end equations into the junction equations

$$P_J = Z_J \sum_{i=1}^{3} \frac{C_{ib}(t-\Delta)}{Z_i} \qquad \frac{1}{Z_J} = \sum_{i=1}^{3} \frac{1}{Z_i} \tag{3.36}$$

and subsequently for the respective flows (*i=1, 2, 3*)

$$Q_{ia} = \frac{1}{Z_i}\left(P_J - C_{ib}(t-\Delta)\right) \tag{3.37}$$

The filtered characteristic pressures at each port are then propagated back as inputs to the connecting models [according to eq.(3.16)] ready for the next time-step, **t+Δ**. The above equations are easily

extended to incorporate any number of ports for different junction models.

There are two arguments in favour of using this method over that proposed by Krus. Firstly, the circuit model can map directly onto the physical topology, which will be of assistance when developing different partitioning arrangements for parallel operation. Moreover, different line-type models (compressibility, long-line etc) may be inter-connected using the line-junction model.

§3.7 Signals

The propagation of information at very high wave speeds, such as electrical signals used in feedback loops, requires a slightly modified approach when using TLM. A typical example might be a valve-controlled actuator with displacement feedback. In this case an instantaneous, uni-directional link must be introduced between the component generating the feedback signal, the control process and the modulating device. The correct component-to-model execution sequence is therefore very important. The fluid lines are still separated numerically by the line delay, but components in the feedback loop are instantaneously linked using *signals*. Burton *et al* [1993A] demonstrate the use of this approach in the simulation of an electro-hydraulic position-control servo-system, using both analogue and digital control.

However, this technique creates problems when partitioning a circuit containing these elements into parallel tasks, owing to the instantaneous coupling between components in the feedback path. The number of possible parallel configurations is thereby reduced, but this limitation may not necessarily impede efficient operation, although this depends upon circuit size and complexity. This subject is discussed more fully in Chapter 6, with reference to a specific case.

§3.8 Closure

In this chapter the basic tools for constructing TLM-based computer simulations of hydraulic systems have been developed. These include:

(i) Hydraulic line modelling using TLM

(ii) Cavitation modelling

(iii) Variable line volume and bulk modulus

(iv) Variable time-steps and physical discontinuity handling

(v) Inter-connection of more than two components to the same fluid volume

(vi) Control signals

These techniques have been implemented into a reconfigurable simulation program, that enables a complete system model to be created from a circuit configuration file and a library of standard components, such as pumps, motors, valves and actuators. The process of circuit model generation using modular components is explained fully in Chapter 5. In the next chapter some standard component model types are developed, which are subsequently employed in the simulation examples.

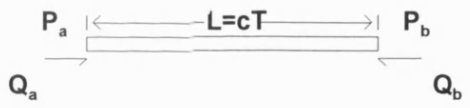


FIGURE 3.1 SYMMETRICAL TRANSMISSION-LINE

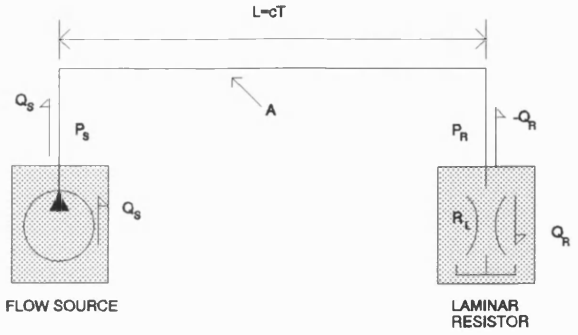


FIGURE 3.2 SIMPLE TLM CIRCUIT

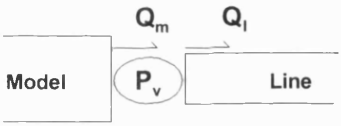


FIGURE 3.3 IDEALIZED VAPOUR CAVITY

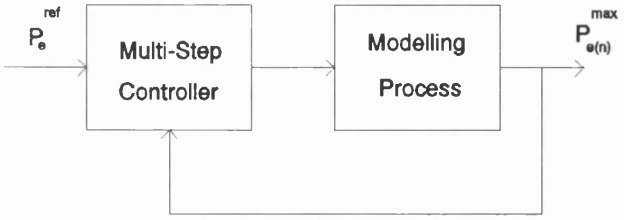


FIGURE 3.4 MULTI-STEP CONTROL PROCESS

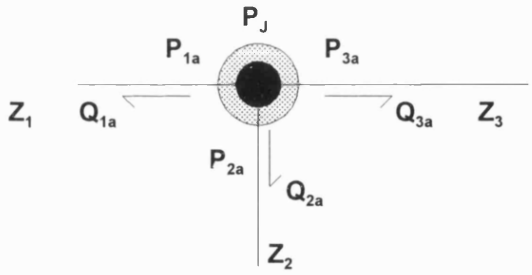


FIGURE 3.5 THREE-PORT TRANSMISSION-LINE JUNCTION

CHAPTER 4

COMPONENT MODELLING

§4.1 Introduction

This chapter serves to illustrate the approach to modelling hydraulic systems and related components in a systematic way; it is not the intention to present details of all the models developed by the author.

Basically, a component model is implemented by solving simultaneously the component equation, or system of equations, with the transmission-line *end* equations at each component port; each transmission line end is incorporated into a separate component model (there are no separate line models as in lumped parameter simulation). Each TLM component model can be represented mathematically as the solution to the following system of equations:

$$\begin{aligned} g(Q) &= h(P) \\ (P - ZQ)(t) &= C(t - \Delta) \end{aligned} \tag{4.1}$$

For systems of components that contain algebraic equations, the solutions to eqs(4.1) are explicit and unconditionally stable [Johns, 1977] [Kitsios & Boucher, 1986]. Each component model is decoupled in time and is thus numerically isolated from its connecting component models by the finite delay introduced by the transmission lines; the solution is computed in discrete time in accordance with the analytical time-domain solution to the loss-less wave equations imbedded within each model.

For components that involve differential equations, a suitable time-difference form can be obtained by numerical integration; in effect this transforms the ODE into an algebraic equation, which is then solved simultaneously with the transmission line ends. Bi-linear transformation, equivalent to trapezoidal integration, is used to solve equations of motion for the mechanical parts of models

reported in this thesis.

Trapezoidal integration is an "A-stable" method, closely related to the integration performance of undamped transmission line modelling [Krus *et al*, 1990]. However, for equations with very short time-constants that must be solved within a component model bounded by fluid-filled transmission lines, numerical integration using the Trapezoidal Rule can prove erroneous; this is because numerical approximations to large eigen-solutions decay only very slowly. Conversely, integration using the Backward Euler method will result in the rapid decay of such eigen-solutions. Both methods exhibit "A-stability", but only Backward Euler has the property of "L-stability" [Lambert, 1973] [Hairer & Wanner, 1991]. Trapezoidal integration has proven to be adequate for most simulations, because when compared with the fluid transients the motion of actuators is often considerably slower dynamically - an observation that applies to hydraulic systems in general.

Solving the equations of motion for the mechanical part of a system together with the hydraulic part can sometimes present problems with respect to component modularity. For example, the transformed differential equations describing a mechanical *load* acting on a component must be solved simultaneously with the hydraulic TLM equations. This necessitates the assimilation of the mechanical load model into the hydraulic component model. An alternative to the hydro-mechanical combined model (and the use of numerical integration to solve the equation of motion) can be provided by the use of mechanical transmission line elements in addition to hydraulic line elements, where force/torque is analogous to pressure and linear/angular velocity is analogous to flow [Partridge *et al*, 1987] [Krus *et al*, 1990]. In this research mechanical transmission line elements were not strictly necessary for the circuit examples studied (combined actuator-load and motor-load models have been used), which simplified the computer programming.

§4.2 Component Modularity

A modular approach forms the basis of simulating a system from generic component models and makes the efficient partitioning of a circuit model into parallel processes much easier to achieve. Modularity is not a problem with many hydraulic component models, such as constant speed pumps, or valves, which are inter-connected only by fluid transmission-lines. However, for components that involve the actuation of a mechanical system, component modularity can be more difficult to achieve. As an example, consider the interaction between separate linear actuator and mass sub-systems, as

in Figure 4.1. The actuator model requires an external force (provided by the external mass-system) to calculate acceleration from the piston mass using Newton's Second law. However, the external mass system (eg. an articulated manipulator) requires an acceleration to determine the applied force. This results in a causal, or algebraic loop, as each model requires these inputs simultaneously. For simple mechanical systems, it is straightforward to refer the effective mass and load force to the actuator model and eliminate the causality by solving the equations for the combined system, shown schematically in Figure 4.2. For highly complex non-linear mechanical loads, such as those associated with linkage systems and manipulators, the use of equations derived for constant effective mass systems may be acceptable, provided that the referred masses and loads are constantly updated by using linkage positions and velocities obtained from a past system state. In many circumstances the sampling delay introduced can be tolerated when referring the mass/inertia terms back to the actuation devices [Krus *et al*, 1991C]. Nevertheless, the geometry and the kinetic/potential energy of the mechanical system are time-dependent, which will effect the accuracy of the mass and applied force terms referred back. In some cases the simultaneous solution of the full non-linear equations for both the load system and actuation device, or devices becomes unavoidable, as in the case of elastic mechanisms [Davison *et al*, 1993].

Another technique is suggested by Krus *et al* [1990] [1991C]. In this approach the hydraulic actuation device is decoupled from the mechanical system according to a *stiff-spring* mechanical transmission-line element, where velocity is analogous to flow and force analogous to pressure, as shown in Figure 4.3. The mechanical system now interacts with the hydraulic system through the propagation of characteristic force waves, or pulses. Although the causality between the hydraulic and mechanical systems has been removed (without solving the combined model), a new dynamic element has been introduced. This element may be difficult to quantify in practice, as it relates to the elastic connection between actuator and load system; a steel-on-steel pin joint, for example. As the elastic element is described by transmission-line equations, it will have an associated inertia of its own, depending upon the simulation time step, in much the same way as the hydraulic transmission-line (§3.2.1). For very stiff distributed springs this may force the time step down to very small values to achieve the correct dynamic response. The difference in applied forces at each transmission-line end becomes an error term used in the variable time step TLM scheme, as the differential force is representative of inertia effects.

A possible alternative is the construction of an inertial line (as opposed to mechanical load model linked via a stiff spring transmission line). In this case the inertia term is kept constant and stiffness (analogous to fluid compressibility) is controlled by varying the simulation time-step; the parasitic stiffness introduced by the inertial line is monitored by measuring the difference between velocity at each end of the mechanical transmission line and controlled by changing the simulation time-step. For the present study the hydraulic actuation device (ie. linear/rotary actuator, or motor) is combined with a simple referred mass, or inertia system and the equations solved simultaneously. This approach does not preclude the future use of separate mass/inertia systems with mechanical TLM element connections, but provides a starting point for the parallel processing investigation.

§4.3 Model Simplification

The equations that describe the behaviour of a component should be subjected to critical evaluation, in order to simplify the component model without significantly affecting the simulated behaviour of the hydraulic system. Frequently, overcomplicated models lead to computational problems. The most detrimental of which is to increase the computation time for no significant improvement in accuracy [Innis & Rexstad, 1983] [Tomlinson, 1987]. Component models may be classified into (i) dynamic models, (ii) semi-empirical models, (iii) pseudo-dynamic models and (iv) instantaneous-dynamic models.

A full dynamic model is derived from *first principles* and involves modelling the individual internal working parts component. Conversely, an instantaneous-dynamic model may be obtained by neglecting internal component dynamics. These dynamics are assumed to be sufficiently fast that they can be treated as instantaneous changes, hence steady-state algebraic correlations between input and output variables are used. A typical example might be a pressure-compensated flow-control valve, a quite satisfactory model may be derived using the empirical flow-pressure characteristic supplied by the manufacturer, based on steady-state measurements. A fully-dynamic model of the working parts is likely to result in a less accurate model, due to the difficulty in obtaining accurate parameter values for the individual valve elements [Tomlinson, 1987].

Handroos [1991] adopts a *semi-empirical* approach that requires some experimental verification before generating a component model. This has merit, because dismantling the component to obtain accurate parameter values is no longer necessary.

It is possible to construct a *pseudo-dynamic* model, by incorporating a typical first or second order response into the instantaneous-dynamic model [Viersma, 1980]. Values for time constants to approximate the valve dynamics may then be estimated and, if required, a parametric study undertaken to determine the influence on the overall system performance.

In this research, instantaneous and pseudo-dynamic models are used as far as possible in order to simplify the system simulations undertaken, but does not exclude the future use of other (more detailed) component model types. The main exception are linear or rotary hydraulic actuators (including motors), where movement is modelled by numerical integration of the equation of motion within the component model.

§4.4 TLM Hydraulic Component Models

Component modelling involves the simultaneous solution of a set of algebraic, or differential-algebraic equations representing the component with the hydraulic transmission-line equations at each hydraulic connection. Seven typical component models will now be described in detail.

§4.4.1 Pressure Source (Reservoir/Tank)

To demonstrate the application of TLM to the most fundamental component type, consider the model of a constant pressure reservoir, shown schematically in Figure 4.4. In this case the constant pressure, P_t, is substituted directly into the transmission-line equation to obtain the flow, Q_t, out of the component, eq.(4.4).

Component equations:

$$P_{1a} = P_t \qquad Q_{1a} = Q_t \tag{4.2}$$

[Note: The first suffix of the line pressures and flows indicates the model port number (only one port in this model) and the second indicates the transmission-line end.]

Transmission line end equation (ie. the end of the transmission line incorporated into the pressure source model):

$$P_a - ZQ_a = C_b(t-\Delta) \tag{4.3}$$

where C_b is the characteristic pressure pulse input from the component connected to the reservoir/tank and Z is the characteristic impedance of the connecting line.

Component flow can be derived explicitly by substitution of the component equations into the general transmission line end equation:

$$Q_t = \frac{1}{Z}\left(P_t - C_b(t-\Delta)\right) \tag{4.4}$$

§4.4.2 Positive Displacement Pumps

Figure 4.5 shows an instantaneous model of a hydraulic pump (connected to a fixed speed prime-mover) that incorporates linear slip and compressibility-flow losses. Linearised losses can be used to characterise the performance of positive displacement pumps over a specified range of operation. (For further information on more complex loss models, see McCandlish & Dorey, 1983).

Pump flow equation, including inter-port leakage and compressibility losses:

$$Q_p = D_p\omega_{pm} - \frac{c_s D_p}{\mu}\left(P_{2a} - P_{1a}\right) - \frac{\left(V_c + D_p\right)}{B_e}\omega_{pm}\left(P_{2a} - P_{1a}\right) \tag{4.5}$$

[Note: the displacement may vary according to an time-varying input demand, or control signal]

Assigning the internal pump flow (positive from port 1 to port 2) to the general transmission line end flows (positive into the transmission line):

$$Q_p = -Q_{1a} = Q_{2a} \tag{4.6}$$

General hydraulic transmission-line equations at each component port:

$$\begin{aligned} P_{1a} &= C_{1b}(t-\Delta) + Z_1 Q_{1a} \\ P_{2a} &= C_{2b}(t-\Delta) + Z_2 Q_{2a} \end{aligned} \tag{4.7}$$

Inserting pump flow into transmission line end equations:

$$\begin{aligned} P_{1a} &= C_{1b}(t-\Delta) - Z_1 Q_p \\ P_{2a} &= C_{2b}(t-\Delta) + Z_2 Q_p \end{aligned} \tag{4.8}$$

Substituting eq.(4.7) into eq.(4.5), to obtain an explicit function for pump flow, in terms of the input characteristic pressures from connected component models (ie. C_{1b} and C_{2b}):

$$Q_p = \frac{D_p \omega_{pm} - \left(\frac{c_s D_P}{\mu} + \frac{V_c + D_p}{B_e} \omega_{pm} \right) \left(C_{2b}(t-\Delta) - C_{1b}(t-\Delta) \right)}{1 + \left(\frac{c_s D_p}{\mu} + \frac{V_c + D_p}{B_e} \omega_{pm} \right) (Z_1 + Z_2)} \tag{4.9}$$

The pump flow is now an explicit function of the input characteristic pressures at the inlet and outlet ports that have been propagated to the component from the connecting component ports of other models. Once pump flow is determined the connecting pressures are calculated directly from eq.(4.8). Consequently, the *filtered* characteristic pressures may be determined according to eq.(3.16) and propagated along the transmission lines to the connecting components to arrive after a finite transmission delay.

A fairly straightforward modification is the uni-directional pump model, which incorporates an integral reservoir. Alternatively, if pump flow losses are not significant, a simple flow source model may be employed. The flow source is derived in a similar manner to the pressure source model, this time using a pre-determined flow to calculate pressure (the example in §3.2 uses such a model).

§4.4.3 Square-Law Orifice Model Restrictors and Directional Control Valves

Consider the turbulent orifice model shown in Figure 4.6, as an example of a highly non-linear component model.

Flow correspondence:

$$Q_o = -Q_{1a} = Q_{2a} \tag{4.10}$$

Square-law orifice equation:

$$Q_o = K\sqrt{\Delta P} \qquad \Delta P = P_{1a} - P_{2a} \tag{4.11}$$

where

$$K = C_D A_o \sqrt{\frac{2}{\rho}} \tag{4.12}$$

or from rated pressure-flow data:

$$K = \frac{Q_{ref}}{\sqrt{\Delta P_{ref}}} \tag{4.13}$$

Flow is determined explicitly by substitution of the following pipe-end *characteristic pressure* equations into the orifice equation.

$$P_{1a} = C_{1b}(t-\Delta) - Z_1 Q_o$$
$$P_{2a} = C_{2b}(t-\Delta) + Z_2 Q_o \qquad (4.14)$$

When the flow is in the positive direction (port one to port two), the orifice flow is obtained from the solution of the resultant quadratic equation:

$$Q_o = -\frac{K^2}{2}(Z_1+Z_2) + \frac{1}{2}\sqrt{K^4(Z_1+Z_2)^2 + 4K^2(C_{1b}(t-\Delta) - C_{2b}(t-\Delta))} \qquad (4.15a)$$

The positive quadratic root has been taken, as the orifice flow must be zero when the differential characteristic pressure is zero. Flow reversal occurs when the differential characteristic pressure becomes negative; this is handled by the following equation, resulting in negative values for orifice flow.

$$Q_o = \frac{K^2}{2}(Z_1+Z_2) - \frac{1}{2}\sqrt{K^4(Z_1+Z_2)^2 + 4K^2(C_{2b}(t-\Delta) - C_{1b}(t-\Delta))} \qquad (4.15b)$$

§4.4.3.1 Numerical Difficulties

Square-law orifice models connected to compressible fluid volumes are well known to cause numerical stiffness problems when modelled using lumped parameter theory, owing to the highly non-linear behaviour of the orifice characteristic flow-pressure equation near the origin. It is for this reason that in most lumped-parameter simulation packages, a linear flow-pressure region is included in the model to remove the stiffness problem and allow the use of much larger time steps in the null flow region. However, a discontinuity point may occur between the linear and non-linear/square-law flow regimes, which forces the integrator to re-start.

The gradient of the orifice flow-pressure characteristic for small flows presents no difficulty for the transmission-line implementation of the square-law orifice model, as the solution is explicit and unconditionally stable. The same version of equations is used throughout the simulation and there is no flow discontinuity.

§4.4.3.2 Directional Controls

Directional control valves can be assembled from the basic transmission-line orifice model, using rated flow-pressure data between connected ports. Figure 4.7 shows an example valve (the external arrow indicates a time varying input demand, or control input for valve spool position). If required,

proportional control of the valve is achieved by assuming a linear variation in the orifice coefficient, K, with opening, corresponding to a linear change in the annular flow area across the valve spool.

§4.4.4 Pressure Relief and Check Valves

A simple relief valve model (shown in Figure 4.8) assumes that the valve dynamics are instantaneous. This valve model incorporates a very significant discontinuity point, as no flow takes place until the differential pressure exceeds the pre-set cracking pressure, P_c. Furthermore, the valve flow is uni-directional. A linear flow-pressure characteristic, of gradient **K**, is then assumed once the valve has opened. This straight-line characteristic is in accordance with most manufacturer's flow-pressure correlations, particularly those relating to two-stage relief valves.

Transmission-line end equations at each valve port:

$$\begin{aligned} P_{1a} &= C_{1b}(t-\Delta) - Z_1 Q_v \\ P_{2a} &= C_{2b}(t-\Delta) + Z_2 Q_v \end{aligned} \tag{4.16}$$

Valve flow to differential pressure:

$$\begin{aligned} (P_{1a}-P_{2a}) > P_c \qquad & Q_v = K(P_{1a}-P_{2a}-P_c) \\ (P_{1a}-P_{2a}) \le P_c \qquad & Q_v = 0 \end{aligned} \tag{4.17}$$

Flow is then determined explicitly by substituting the pipe-end (characteristic pressure) equations into the valve equation, as follows:

$$\begin{aligned} P_{1a}-P_{2a} > P_c \qquad & Q_v = K\left(C_{1b}(t-\Delta) - C_{2b}(t-\Delta) - (Z_1+Z_2)Q_v - P_c\right) \\ & \therefore \; Q_v = \frac{K(C_{1b}(t-\Delta) - C_{2b}(t-\Delta) - P_c)}{1+K(Z_1+Z_2)} \\ P_{1a}-P_{2a} \le P_c \qquad & Q_v = 0 \end{aligned} \tag{4.18}$$

The respective valve pressures are calculated by substitution of the valve flow into the transmission line end equations at each port. Hence, the characteristic pressures at each port are determined and transmitted to connecting components.

A sudden change in valve flow due to the sudden discontinuity at the cracking pressure, will be detected as a significant pressure *error* in the connecting transmission-lines by the time step (error) controller. Consequently, the time step will be adjusted until the discontinuity point has been located

to the level of accuracy specified by the system modeller (§3.5).

This model is equally applicable to check valves.

§4.4.4.1 Pilot Operated Valves

The analysis above also holds for pilot-operated relief valve and pilot-operated check valve models. In this case the valve opening condition is modified to account for pilot operation from a third hydraulic connection, instead of from the valve differential pressure alone. When the pilot pressure exceeds the pre-set valve cracking pressure, for example, the valve opens and the calculation proceeds, as before.

The pilot port is usually modelled as a hydraulic transmission-line with a blocked end at the valve, which necessitates zero flow at the line termination, meaning the pilot pressure is equal to the characteristic pressure. The zero flow condition is a reasonable modelling assumption, as the flow at the pilot port will be negligible. Alternatively, a small leakage path may be incorporated, depending upon valve construction and the level of accuracy required.

§4.4.5 Positive Displacement Motors and Loads

Figure 4.9 shows schematically the combined motor and inertial load model, with an externally applied load torque duty-cycle input. The hydraulic motor incorporates linearised loss coefficients in a similar manner to the linear-loss pump model (for the motor model the inter-port leakage is positive).

Ideal flows plus leakage flow:

$$Q_m = D_m \omega + \frac{c_s D_m}{\mu}(P_{1a} - P_{2a}) \tag{4.19}$$

Substitute transmission-line equations for port pressures:

$$Q_m = \frac{D_m \omega + \frac{c_s D_m}{\mu}(C_{1b}(t-\Delta) - C_{2b}(t-\Delta))}{1 + \frac{c_s D_m}{\mu}(Z_1 + Z_2)} \tag{4.20}$$

Motor torque including losses:

$$T_m = D_m(1-c_f)(P_{1a}-P_{2a}) - c_v D_m \mu \omega \tag{4.21}$$

Substitute transmission-line equations for port pressures:

$$T_m = \frac{D_m(1-c_f)(C_{1b}(t-\Delta)-C_{2b}(t-\Delta))}{1+\frac{c_s D_m}{\mu}(Z_1+Z_2)} - \left(\frac{D_m^2(1-c_f)(Z_1+Z_2)}{1+\frac{c_s D_m}{\mu}(Z_1+Z_2)} + c_v D_m \mu\right)\omega \tag{4.22}$$

Inertial load equation:

$$J\dot{\omega} = T_m - T_L(t) - T_R\frac{|\omega|}{\omega} - c_L\omega \tag{4.23}$$

where T_R and c_L are the resistive torque and load damping rate respectively.

The connecting shaft speed is obtained by substitution of eq.(4.22) into eq.(4.23) and re-arranging:

$$J\dot{\omega} = \frac{D_m(1-c_f)(C_{1b}(t-\Delta)-C_{2b}(t-\Delta))}{1+\frac{c_s D_m}{\mu}(Z_1+Z_2)} - \left(\left(\frac{D_m^2(1-c_f)(Z_1+Z_2)}{1+\frac{c_s D_m}{\mu}(Z_1+Z_2)}\right)+(c_v D_m)\mu\right)\omega - T_L(t) - T_R\frac{|\omega|}{\omega} - c_L\omega \tag{4.24}$$

An explicit form of motor shaft speed for digital computation is obtained by substituting the bi-linear transform, equivalent to the trapezoidal rule for integration, from which the other model parameters can be determined. This process is explained as follows.

The bi-linear transform applied to motor speed (allowing for variable step length) is:

$$\omega_{n+1} = \frac{\Delta_n}{2}(\dot{\omega}_n + \dot{\omega}_{n+1}) + \omega_n \tag{4.25}$$

Solving the resultant implicit equation for motor speed:

$$\omega_{n+1} = \frac{\frac{\Delta_n}{2J}\left(f(C_{1b}(t_n-\Delta_{n-1})-C_{2b}(t_n-\Delta_{n-1})) - T_L(t_n) - T_R|\omega_n|\right) + \left(\frac{\Delta_n}{2}\dot{\omega}_n + \omega_n\right)}{1+\frac{\Delta_n}{2J}\left(fD_m(Z_1+Z_2) + D_m c_v \mu + c\right)} \tag{4.26}$$

where

$$f = \frac{D_m(1-c_f)}{1+(Z_1+Z_2)\frac{c_s D_m}{\mu}}$$

§4.4.5.1 Rotary Actuator

It is a relatively straightforward process to extend the above analysis for a rotary actuator, simply by integrating angular speed to obtain displacement and setting appropriate travel limits. The physical discontinuity at either end stop is manifested by a corresponding discontinuous flow at the model port, as speed is set to zero upon reaching the travel limit. This condition is detected as a significant pressure error in the connected transmission-lines, causing the variable time step controller to reject successively smaller steps according to the step-size control algorithm used [eq.(3.30)], until the discontinuity (high, or low travel limit) is located to the specified accuracy.

§4.4.5.2 Gear Reduction

Torque amplification can be achieved using a speed reduction ratio between a hydraulic motor and the driven load. The load inertia, viscous damping, coulomb friction and load torques are referred to the motor shaft and added to the corresponding parameters at the motor. In addition, a constant mechanical efficiency may also be incorporated into the model to account for frictional losses.

For a reduction ratio, n (and neglecting friction losses):

$$\begin{aligned} J &= J_m + \frac{J_L}{n^2} \\ c &= c_m + \frac{c_L}{n^2} \\ T_R &= T_{R,m} + \frac{T_{R,L}}{n} \\ T &= \frac{T_L}{n} \end{aligned} \tag{4.27}$$

where subscript m refers to the motor side of the gearbox and L refers to the load side.

§4.4.5.3 Flow-divider

The flow-divider, as seen in the circuit example of Figure 2.1, can be modelled by treating it as two hydraulic motors coupled by an inertial load. The equations of motion for both motors and the load can then be solved simultaneously. Alternatively, an elastic *stiff spring* interface element (torsional transmission-line) may be employed to link two separate motor models together via a separate inertial load model [Partridge *et al*, 1987]. This is solved, as before, using the bi-linear transform. The now separate motor models then have three transmission-line connections in total, two hydraulic and one

torsional. The torsional element is of the form:

$$T - Z\omega = C_{\omega}(t-\Delta)$$
$$\left[Z = \Delta\left(\frac{GJ}{L}\right)\right] \tag{4.28}$$

where the function of Z in brackets represents the torsional stiffness of the connecting shaft, in torque per unit angular deflection.

As discussed in §4.2, component modularity is increased, at the expense of modelling additional and possibly unnecessary dynamics. Error control of the stiff shaft (to ensure it does not become "distorted" with too much inertia) may be included by monitoring the difference in torque at each end of the torsional transmission-line. Nevertheless, very stiff shafts can force small simulation time steps for significant changes in rotational speed.

§4.4.6 Differential Area Linear Actuator and Loads

This model of a differential area actuator incorporates a combined mass system, as shown in Figure 4.10, which includes the integration of two internal state variables: velocity and displacement. Non-linear effects include the presence of slip-stick friction, contact of the load with a linear spring, or the piston reaching an end-stop. The variable fluid volumes either side of the piston are accounted for by adding the volume to the connecting transmission-line volumes, altering their compressibility (§3.4). In addition, the load force may be an external time-varying input demand.

Transmission-line equations at the piston:

$$P_1 = C_{1b}(t-\Delta) - Z_1 Q_1$$
$$P_2 = C_{2b}(t-\Delta) + Z_2 Q_2 \tag{4.29}$$

Piston and rod end flow equations:

$$Q_1 = vA_1$$
$$Q_2 = vA_2 \tag{4.30}$$

Piston equation of motion

without spring force ($x<x_K$):

$$M_e \dot{v} = P_1 A_1 - P_2 A_2 - F_L(t) - F_R \frac{|v|}{v} - c_L v \tag{4.31}$$

including spring force ($x>=x_K$):

$$M_e \dot{v} = P_1 A_1 - P_2 A_2 - F_L(t) - F_R \frac{|v|}{v} - c_L v - K(x - x_K) \tag{4.32}$$

where F_L, c_L refer to the externally applied load force and load damping coefficient respectively. F_R refers to the resistive (coulomb friction) force that opposes motion.

Displacement:

$$v = \dot{x} \tag{4.33}$$

Solving the transmission-line, flow and relevant equation of motion, the following equations are obtained.

Without spring force ($x<x_K$):

$$M_e \dot{v} = C_{1b} A_1 - C_{2b} A_2 - \left(Z_1 A_1^2 + Z_2 A_2^2 + c_L\right) v - F_L(t) - F_R \frac{|v|}{v} \tag{4.34}$$

With spring force ($x>=x_K$):

$$M_e \dot{v} = C_{1b} A_1 - C_{2b} A_2 - \left(Z_1 A_1^2 + Z_2 A_2^2 + c_L\right) v - F_L(t) - F_R \frac{|v|}{v} - K(x - x_K) \tag{4.35}$$

Depending upon the region of operation, the above equations of motion are solved using bi-linear transformation (trapezoidal integration) to obtain a pair of discrete time-difference equations (with and without the spring force). This discontinuity may be detected and tracked by the step controller, again because of the effect on flow transients and hence the pressure differentials in the connecting transmission lines.

The discrete-time equations for piston velocity are obtained by substitution of the following bi-linear transforms, and the solution of the resulting implicit equations:

$$v_{n+1} = \frac{\Delta_n}{2}\left(\dot{v}_n + \dot{v}_{n+1}\right) + v_n \tag{4.36}$$

$$x_{n+1} = \frac{\Delta_n}{2}\left(v_n + v_{n+1}\right) + x_n \tag{4.37}$$

Piston velocity without spring force:

$$v_{n+1} = \frac{\dfrac{\Delta_n}{2M_e}\left(C_{1b}(t_n - \Delta_{n-1})A_1 - C_{2b}(t_n - \Delta_{n-1})A_2 - F_L(t_n) - F_R \dfrac{|v_n|}{v_n}\right) + \left(\dfrac{\Delta_n}{2}\dot{v}_n + v_n\right)}{1 + \dfrac{\Delta_n}{2M_e}\left(Z_1 A_1^2 + Z_2 A_2^2 + c\right)} \tag{4.38}$$

Piston velocity with spring force:

$$v_{n+1} = \frac{\frac{\Delta_n}{2M_e}\left(C_{1b}(t_n-\Delta_{n-1})A_1 - C_{2b}(t_n-\Delta_{n-1})A_2 - F_L(t_n) - F_R\frac{|v_n|}{v_n} - K\left(\frac{\Delta_n}{2}v_n + x_n - x_k\right)\right) + \left(\frac{\Delta_n}{2}\dot{v}_n + v_n\right)}{1 + \frac{\Delta_n}{2M_e}\left(Z_1A_1^2 + Z_2A_2^2 + c + K\frac{\Delta_n}{2}\right)} \tag{4.39}$$

§4.4.6.1 Stick-slip Modelling

A *stiction* force often exists in the moving parts of a component, such that a high break-away force, or torque is required. A constant and usually much smaller force (the coulomb friction) resists motion once the initial stiction force is exceeded. In terms of the component models, such as the combined actuator-mass model, this is represented by the following inequalities:

$$|F_{net}| \le |F_s| \quad \& \quad |v| \le v_{tol} \quad : \quad v = \dot{v} = 0 \tag{4.40}$$

In order to "stick", the piston velocity must be within a specified tolerance band v_{tol} about zero (in this research a value of 1mm/s has been taken). Furthermore, the net applied force F_{net} (including pressure forces, external load force and gravitational terms) must be less than or equal to the stiction force F_S. If both conditions are satisfied then piston acceleration and velocity are set to zero. In order to "slip", ie. for piston acceleration to be computed according to its equation of motion, then F_{net} must exceed F_s.

The phenomenon of stick-slip is a very significant non-linearity, responsible for highly oscillatory start-up transients in actuation devices, which generally force the error control process to use very small time steps in order to follow it accurately.

§4.4.7 Gas-type Accumulator

The final model to be considered here is the gas-type hydraulic accumulator model, shown in Figure 4.11. This is particularly interesting, due to the non-linearity associated with the gas expansion and compression process. The following equation gives the initial *charged* accumulator gas volume, based on the system gas volume (equal to the maximum accumulator gas volume V_a and the volume of any additional back-up bottles V_{cyl}), the accumulator pre-charge pressure P_a and the charged pressure P_{ch}. The charging operation is assumed thermodynamically slow.

For the isothermal gas process during charging:

$$V_{ch} = \frac{(V_a + V_{cyl}) P_a}{P_{ch}} \tag{4.41}$$

The transmission-line equation at gas-liquid interface (typically a flexible diaphragm, or membrane seal) is given by the following equation, assuming that gas pressure equals liquid pressure and gas flow equals liquid flow:

$$P_g - ZQ_g = C_b(t - \Delta) \tag{4.42}$$

where Z is the line impedance of the connecting (liquid filled) transmission line. [Note: in this model the membrane between gas and liquid has negligible inertia and negligible stiffness.]

For the polytropic gas process during charging and discharging in normal operation:

$$P_g = \frac{P_{ch} V_{ch}^{\ n}}{V_g^{\ n}} \tag{4.43}$$

Because the accumulator dynamics are relatively slow, it is sufficient to use the previous value of gas volume V_g to determine the gas pressure from eq.(4.43). Hence flow at the gas-liquid interface can be obtained from eq.(4.42). The updated gas volume is obtained by integrating the gas flow using the trapezoidal rule. The volume of trapped fluid in the accumulator body is added to the connecting transmission-line as the accumulator charges and discharges with oil.

Testing of this model by the author revealed this to be a satisfactory approach, with modelling errors for volume typically less than two percent. The alternative is to solve the resulting polynomial using an iterative solver, such as Newton-Raphson, every time step. This approach can be less reliable, owing to limitations of the Newton-Raphson solver, and often leads to some variation in model run-time because the number of iterations to convergence can vary.

This model could be enhanced by incorporating thermal effects in the accumulator model equations [Harris, 1990]; thermal transients can be solved by numerical integration within the accumulator model (Harris [1990] uses Euler integration).

§4.5 Cavitation Modelling

Individual component models handle cavitation by checking the pressure at each hydraulic

transmission-line connection. The vaporous cavitation algorithm detailed in Chapter 3 (§3.3) is initiated if the pressure falls below the fluid vapour pressure. This phenomenon will require small time steps to locate the discontinuity regions and follow the subsequent pressure transients properly.

§4.6 Closure

In Chapter 4 several important hydraulic component models have been developed using the transmission line modelling technique, illustrating the modular approach adopted for simulation. However, in all but the simplest component models, the TLM approach can result in rather complex (and not easily recognised) algebraic equations, eg. the motor-load model. The development of such models might be aided by the use of symbolic algebra manipulation software, such as *Mathematica* from Mathworks inc, to reduce the likelihood of errors in component model formulation.

The TLM components developed, taken in conjunction with the techniques explained in the previous chapter, has enabled the development of an automatic *code generation* program to simplify and improve the reliability of the modelling process. This process is described in the following chapter.

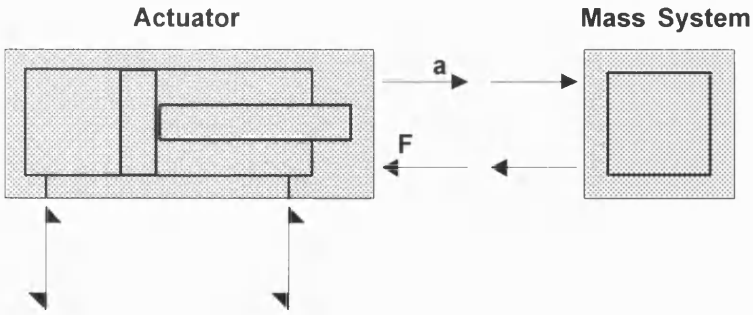


FIGURE 4.1 ACTUATOR / MASS SYSTEM ALGEBRAIC LOOP

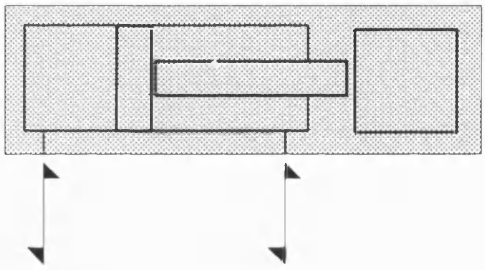


FIGURE 4.2 COMBINED ACTUATOR AND REFERRED MASS SYSTEM

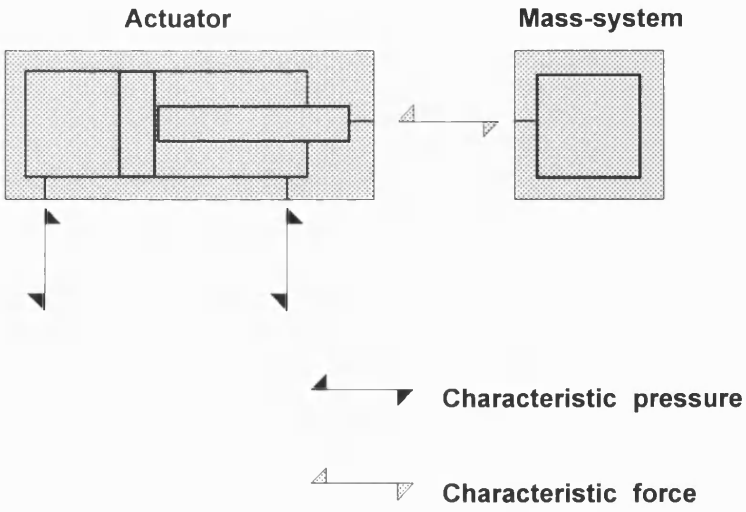


FIGURE 4.3 ACTUATOR MASS-SYSTEM SEPARATED BY A STIFF-SPRING MECHANICAL TRANSMISSION-LINE

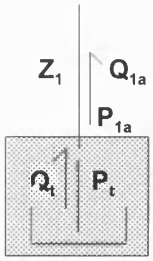


FIGURE 4.4 TLM CONSTANT PRESSURE RESERVOIR

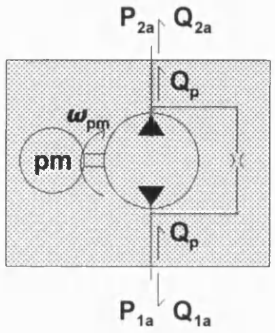


FIGURE 4.5 TLM LINEAR-LOSS PUMP

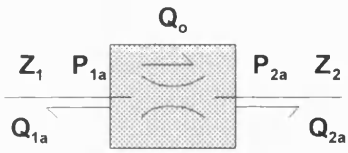


FIGURE 4.6 TLM SQUARE-LAW ORIFICE

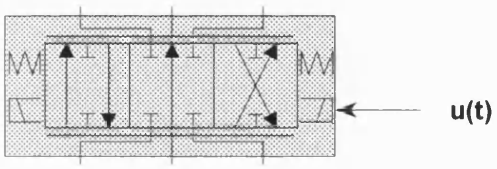


FIGURE 4.7 TLM 6-PORT, 3-POSITION DIRECTIONAL CONTROL VALVE (DCV)

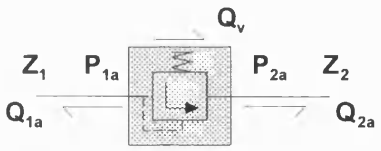


FIGURE 4.8 TLM RELIEF VALVE

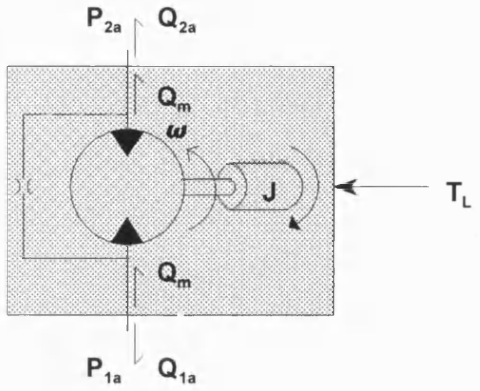


FIGURE 4.9 TLM COMBINED MOTOR-LOAD

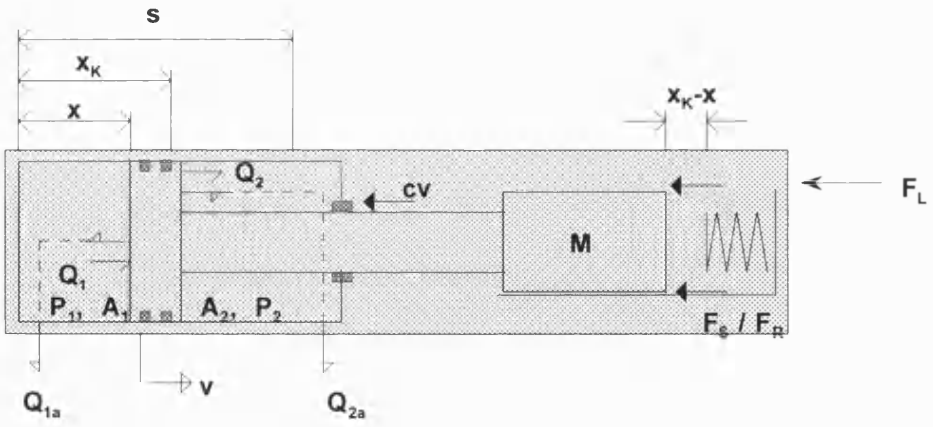


FIGURE 4.10 TLM COMBINED ACTUATOR-LOAD

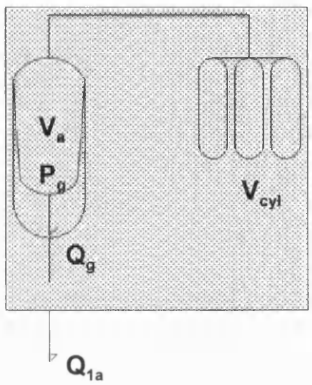


FIGURE 4.11 TLM GAS-TYPE ACCUMULATOR

CHAPTER 5

SYSTEM MODELLING

---

§5.1 Introduction

In this chapter, the process of automated program generation using pre-compiled component models to represent the separate circuit elements, is described for the development of single-processor TLM simulations (for use on the host processor of the **Sun 4/370**). To illustrate this approach, the "two-actuator" circuit first shown in Figure 2.1 is used as a case study. Verification of the predicted results was achieved using a standard package which employs an equivalent system of ODE's, solved using the Adams/Gear integrator "LSODA" [Petzold, 1983] [Richards *et al*, 1991]. The package has been thoroughly tested against experimental data and hence provides a suitable "control". However, it should be noted that the TLM algorithm can include additional wave propagation effects in the lines that are unmodelled by the lumped parameter solver. The relative computational speed of the two different numerical techniques is also compared.

§5.2 Automatic Program Generation

Automated generation of the simulation program enables the system modeller to develop large and complex circuit configurations from much simpler modular elements; the approach adopted by the author for the development of the TLM code generator adopts similar principles to conventional lumped parameter simulation packages, such as "**BATH*fp***" [Richards *et al*, 1990] and "**HASP**" [Tomlinson, 1987].

For the TLM code generator the models used in the simulation and how they are connected together

are defined by the user in an ASCII[4] text file, subsequently referred to as a *link* file. This file is usually written with the assistance of a *link diagram*; a manually drawn sketch, detailing circuit layout in terms of model names and link numbers that represent connections between models. (The concept of a *link* file is similar to that used in the simulation package **HASP**).

The completed *link* file is subsequently interpreted by the TLM automatic *program generator*, which checks the consistency of the circuit defined, writes a control program and links together the relevant component models (the modeller must insert these into the *link* file in the correct sequence for cases involving uni-directional signals, §3.7).

In general, the use of automatic program generation leads to improvements in:

(i) Efficiency. The time involved in building simulation programs is reduced greatly. Modifications to the circuit configuration can then be achieved by altering the simplified circuit description in the system model *link* file.

(ii) Reliability. New component models may be tested in isolation for correct operation, before addition to the model database. Circuits of significant complexity can then be developed with increased confidence, facilitated by linking together validated model objects.

(iii) Simplicity. The analyst is separated from the task of creating models and writing source code directly using a 4th generation language, such as **C**, or **FORTRAN**. Only the development and validation of new models is a detailed and time-consuming task.

Notwithstanding the potential benefits indicated, the development of a *program generator* is a detailed and complex task. The *program generator* and all TLM component models have been written by the author in the **C** programming language using the **Sun 4/370** operating system **SunOS 4.1**. (The **HASP** program generator for example was written entirely in **FORTRAN 77**, and was designed to generate **FORTRAN 77** simulation code).

[Note: *Burton 1994* gives a detailed listing of the (host processor) code generator and the operations required to generate automatically the code used in the TLM simulation example discussed later in

[4] American Standard Code for Information Interchange

the chapter. The following description outlines the main points of operation.]

§5.2.1 Simulation Process

Figure 5.1 shows schematically the complete simulation process. Firstly, the user-supplied *link* file (of the form "*filename*".**link**) is interpreted by the code generation program in three stages, corresponding to three different sections of the *link* file:

(i) System parametric data. This first section of the *link* file contains data on fluid properties, simulation start and end times, maximum pressure error and results print interval.

(ii) Transmission-line data. Here the program generator reads in the individual link numbers and the link type (either hydraulic transmission line, or uni-directional signal). For hydraulic transmission line links, the pipe diameter, length, effective bulk-modulus and initial pressure is also recorded by the program generator.

(iii) Component model data. In this final section the following information is detailed for every component model used in the circuit: the unique model name, the correspondence between model port connections and transmission-line/signal link numbers, followed by a list of component parametric data. The parametric data is entered into the *link* file in a specific sequence that corresponds with the parameter argument list for the model.

When interpreting the *link* file the TLM program generator makes a number of checks to ensure that a valid circuit model can be created. The checking procedure greatly improves the reliability of the modelling process, and ensures that:

(i) Transmission-line and signal links are defined and are connected at both ends to different component models and that the transmission-line/signal link references are not duplicated elsewhere in the circuit.

(ii) Valid parametric data is specified; only non-zero transmission-line lengths are accepted, for example.

(iii) Component models referenced actually exist in the model database (by cross-referencing information obtained from the "**model.attributes**" file) and that all

component ports have a corresponding transmission-line, or signal link reference. The correct numbers of real and/or integer parameter data for each model case is also checked.

For numerical simulations many of the source files written by the program generator have similar structures and many common functions (procedures). It is for this reason that *template* files are read by the program generator line-by-line and used to construct the simulation program *source*, *header* and *make* files, by inserting relevant code excerpts into the file templates. (The use of template files as a means of simplifying the code generation process is not uncommon, and has been adopted in both **BATH*fp*** and **HASP** program generators).

In the *source* program (ie. the top-level simulation control code) the appropriate model call argument list must be written for every occurrence of a component, corresponding to each model entry in the *link* file. Moreover, simulation data associated with each component model must be stored by the control program during execution to ensure data integrity. This is necessary, as there may be multiple occurrences of the same model; it is very likely that more than one relief valve model is used in a circuit, for example, using the same component model code, but having different data.

The automatically-generated *header* file contains only very specific variable declarations, accessible by all sub-programs in the *source* code (ie. global data statements). Finally, the system *make* file contains all of the arguments necessary to build an executable file from all of the constituent elements, ie. the *source* file, *header* file and all of the pre-compiled component models used in the specific circuit model simulation. The *make* file is automatically invoked upon the successful generation of the system model *source* and *header* files by the program generator. A single executable file of the form "*filename*.**ex**" is produced which can be executed on the command line within the **SunOS** (Unix) operating system on the host computer; the filename argument is the same as in the "*filename*.**link**" file first supplied to the program generator.

Following completion of the simulation a binary *results* file is written (of the form "*filename*.**results**"), containing simulation data saved by the separate component models during the simulation. In a fully-integrated graphical environment, the user would simply select interactively the relevant component model icon from an on-screen graphical representation of the circuit and

choose the parameters required for plotting. At the present time, however, each model writes an entry into an **ASCII** reference file during the initialisation phase of the simulation (this file is given the name "*filename*.**ref**" by the program generator and referred to here as a *ref* file). The *ref* file details every component model used in the circuit, the parameters saved by each model for user output (ie. for plotting on-screen) and a unique integer number corresponding to parameter position in the binary results file, including a description of the model the parameter came from and the model "instance" number; this is a unique integer number that corresponds to the models position (from first to last) in the *link* file written by the modeller. The use of a graphical post-processing utility in conjunction with the reference file enables the system modeller to inspect the simulation results.

§5.2.2 Parallel Processing

Conceptually, the implementation of the TLM solution technique in parallel is straight-forward to achieve. Separate component models or sub-circuits of component models can be partitioned on to separate processors. The communication between sub-circuit partitions can take place at the end of each solution step by the transfer of characteristic pressures. This kind of parallel algorithm is sometimes referred to as "domain decomposition", because the component models which describe the system can be mapped onto separate processors directly. The functional detail of the parallel algorithm is discussed in the following chapter (§6.2).

The process of creating a system model by linking individual components on a single processor should also take account of the following general computational requirements to enable efficient multi-processor operation. The ability to use modular code elements, which for the most part can operate independently of the host computer platform (the **Sun 4/370** in this research), is an important factor in the development of an efficient parallel implementation (in this case the **Transtech MCP1000** motherboard and associated **T800** transputers connected to the **Sun 4/370**). Accessing data on the host computer's hard disk from a remote processing node is potentially very time consuming. Consequently, all parametric data is included in the generated simulation program to avoid reduction in execution speed, because of the latent delay in accessing data stored on hard disk, as opposed to memory. This requirement also applies to the generation of results, which are stored in memory (subject to memory size limitations) until the completion of the simulation run, when they are saved to the host computer's disk for later inspection. Making the maximum use of local memory is

important for parallel computing, because separate processors cannot access the disk simultaneously during program execution; this also applies to simulation on the host computer's processor.

§5.3 TLM Numerical Algorithm

The key part of a TLM simulation is the algorithm by which the TLM models are solved. The algorithm developed and used here is illustrated by the flow-chart of Figure 5.2. The algorithm draws on two basic sets of equations - those which define the propagation of information between component models (described in detail in chapter 3), and those which form the component models (described in chapter 4); these equations are not repeated here.

The first stage of a simulation is to establish various global parameters, and those parameters which require initial values. A simulation loop is then entered until the simulation is completed. The first part of the simulation loop uses each component model to predict pressures and flows at each end of each line, from the known characteristic pressures (determined at the previous time-step). For a rapid simulation the time step is adjusted to be as large as possible, whilst maintaining the instantaneous pressure differential across the loss-less lines below some user-defined limit (the reference pressure error). Therefore before accepting the results of these initial calculations the solver checks the pressure differential across each line. If the pressure differential for each line is greater than the reference pressure error then the step is rejected. The time interval is reduced according to the relationship of eq.(3.30) and the step repeated. Conversely if the pressure differential for each line is less than the reference pressure error then the step is accepted. However, the time interval for the next step may be adjusted according to eq.(3.29) (thereby ensuring a rapid rate of solution).

When the step is accepted the transmission line impedances must be recalculated in readiness for the next step, using eq.(3.12). At this point a low-pass filtering algorithm is also applied to the characteristic pressures using eq.(3.16). This filtering algorithm introduces an additional modification to the line impedance. The characteristic pressures must also be compensated to allow for the change in impedance, using eq.(3.27).

The procedure above must allow for the total volume of fluid associated with each line, and this may include fluid contained within the components. However, as this volume is subject to change it is important that the total line volume is recomputed at the start of each loop, and that at the end of each loop the line volumes are reset to the pipe volumes only.

The final part of the loop is to transfer the newly calculated characteristic pressures between connected components. After this results may be stored to an external file, if the time has advanced beyond the last storage time by the user-defined print interval. The simulation time is then incremented by the time-step and, if the end-time has not been reached, the loop starts again.

§5.4 Simulation Example: Two-actuator Circuit

This example compares the TLM simulation of an actuator circuit with its ODE counterpart. There are necessarily some differences in the results from the simulations (for example because a lumped parameter pipe model must have the same pressure at both ends), and it is difficult to say which model is most accurate, in the absence of verified experimental data. However, it is important to examine how the user can affect the results of a TLM simulation (by changing the reference pressure error), and how the speed of a TLM simulation compares with that of a simulation based on ODE models. The purpose of this thesis is to investigate the TLM approach as a means of achieving faster, parallel, simulations.

The two-actuator circuit, as shown previously in Figure 2.1, is an ideal example with which to demonstrate the application of the TLM algorithm. Briefly, the purpose of this hydraulic system is to use the coupled motors to divide the flow equally between each of the two actuators, both during extension and retraction. The two pumps initially operate together, supplying maximum flow to the actuators simultaneously. However, once a pre-defined pressure is reached the larger capacity pump is unloaded, via the pilot operated relief valve. The smaller pump continues to supply flow independently at the higher pressure until either the directional control valve position is altered, or the high pressure relief is activated. This arrangement is intended to limit the maximum power required from the prime mover. A potential drawback of this circuit design is the inability of the flow-divider to supply precisely equal flows. Different leakages in the separate gear motors, which constitute the flow-divider, result in different supplied flows. It is the function of the relief valves connected to the actuator piston ends to facilitate re-synchronisation of position, should one actuator stop before the other.

§5.4.1 System Model Generation

In order to construct the circuit model, it is necessary to define the individual component models and

how they are linked together. Figure 5.3 shows the equivalent *link diagram* of the two-actuator circuit. The terms in parentheses refer to the component model names given in the model index table **model.attrib** (given in detail in *Burton 1994*). The different link numbers refer to the transmission-lines, which connect the models. Table 5.1 (to be read in conjunction with Figure 5.3) gives the corresponding system and model parameters used for this particular simulation example. Further information is given in *Burton 1994*, which details the specific user-defined *link* file **act2.link**, used in the generation of the single processor simulation. The automatically generated **C** code files **act2.c**, **act2.h** and **act2.make** are also given in this report; in addition the model attributes file, **model.attrib**, and the program generator template files used in the automatic generation of the **C** code files are detailed in *Burton 1994*.

§5.4.2 TLM-ODE Comparison

The transmission-line modelling approach was tested against the results of a lumped-parameter circuit simulation, performed using the ODE solver, LSODA. It must be remembered that the lumped parameter line models used by the ODE solver are of the purely capacitive type, as indicated by the following equation.

$$\frac{dP}{dt} = \frac{1}{C}\sum_{i=1}^{n} Q_i \qquad C = \frac{V}{B_e} \tag{5.1}$$

where a positive value of Q_i represents a flow into the fluid volume, V.

TLM on the other hand includes distributed inertia and compressibility in these fluid volumes because of wave propagation effects. These effects can be physically realistic, provided the lines do not become excessively distorted by the TLM solver. The line length of each pipe should not become unrealistically long in comparison with its cross-sectional area in order to maintain the same capacitance. The TLM results tend towards the ODE results for short lines, which may not be a problem provided the modelling of inertia effects is not required by the user. Hence, although the ODE solution is used as a "control" it is not an absolute measure of the accuracy of the TLM solution with the physical system, but more of a guide to performance provided the two solution sets are sufficiently close to be within acceptable modelling errors; a mathematical model can never be an exact representation of the real system, neither can exact measurements be taken from the system.

§5.4.2.1 Results Comparison

Figures 5.4a to 5.4f show a selection of computed results obtained from both the lumped parameter and the transmission-line simulations. A transmission-line transient pressure difference (error) of 0.1 bar and a 1μs minimum time step were used for the TLM solution; these parameters give TLM results nearly identical to the lumped parameter case. Smaller tolerance values for pressure error and minimum time step did not significantly change the TLM solution (although it approached the LSODA solution more closely), but did increase the simulation run time (see Table 5.2).

Figure 5.5a and Figure 5.5b show the magnitude of the time step and the corresponding maximum transmission-line pressure difference used in the variable time step TLM simulation. To increase simulation efficiency, only simulation steps that result in a transmission-line pressure error in excess of 1 bar were rejected and repeated with a much smaller time step. However, the nominal 0.1 bar error tolerance specified is still used in the control laws that govern a change in time-step size following step acceptance. This explains the pressure differences above 0.1 bar that are evident in Figure 5.5b. Moreover, if the error controller requires a time step less than the 1μs minimum allowable, this will also cause spikes in the maximum pressure error transient. The process of step rejection can increase significantly computer run-time for simulations that result in many rejected steps, eg. in the case of very small tolerances (less than 0.01 bar).

Figure 5.4a shows the variation in the simulated actuator position with time, for both actuators. There were negligible differences between the TLM and lumped parameter solution techniques. The differences in displacement, as explained previously, are due to the different leakages incorporated in the two coupled motors (the flow-divider) and hence the supplied flow to the two actuators is not identical. The most significant change in gradient is a velocity change, as a result of the flow redirected to the reservoir from the primary pump, when the pilot operated relief (unloading) valve is opened. Figures 5.4b & 5.4c illustrate the corresponding actuator velocities.

The pilot and inlet pressure to the pilot operated unloading valve is shown in Figure 5.4d. This shows an initial start-up transient and the small pressure differential between the pilot and inlet pressure, due to losses modelled in the check valve. The pilot pressure steadily increases above that of the cracking pressure, as a result of the linearly increasing spring load force. The lumped parameter pressure transients are initially slightly greater than the TLM transients, owing to the small levels of fluid friction modelled by the lumped parameter system; a pseudo-dynamic laminar, or

turbulent line friction model is used, based upon well known steady-state pipe friction correlations. The reduction in flow after unloading the primary pump reduces this effect. For completeness, the flow-divider shaft speed and the actuator piston and rod end pressures, computed using both techniques, are also shown in Figures 5.4e & 5.4f respectively.

It is interesting to compare the relative differences between the two simulation methods. Figures 5.6 show the relative difference curves for some salient transients. Figure 5.6a indicates the pressure difference between the two methods at the unloading valve inlet, normalised against the lumped parameter solution. The maximum relative difference, of approximately 2.5 percent, occurs at the physical discontinuity where the actuators engage the stiff-spring load at 1.7m extension. The mean deviation evident in the pressure transients is, in fact, due to the inclusion of fluid friction in some of the lumped parameter pipe models. This inclusion is unavoidable for the lumped parameter system owing to model linking requirements: in the lumped parameter system all components, including compressible lines, are treated as separate models. Different components, such as valves and actuators, are then linked via compressible line models. To connect the service port of a valve to an actuator via a compressible line model, for example, requires that both the actuator and the valve supply flow and receive pressure from the line, which is integrated centrally from eq.(5.1). Therein lies a numerical inconsistency. The lumped parameter actuator model has internal variable volumes (variable volume compressible lines) at the piston and rod ends, that also require flow. The only means available to overcome this algebraic loop is to introduce an orifice model at the actuator end of the line model (approximating the effect of fluid friction), which computes flow from differential pressure. If line friction is not significant this can be unfortunate, as numerical stiffness can increase substantially when very small amounts of friction are specified (§4.4.3). TLM, by its very nature, has no such linking problems, as all communication between models is handled by the propagation of characteristic pressures at a finite speed.

Returning to Figure 5.6a the steady-state pressures are identical, because there is zero flow and therefore no frictional losses. Similar effects are observed in Figure 5.6b, which shows the normalised pressure difference for the piston end pressure in actuator 1. The lower actuator piston velocity presented by the lumped parameter approach (Figure 5.4b) and therefore flow, corresponds to reduced pressure losses in the lumped parameter simulation because of the necessary, but small amount of fluid friction modelled. Figure 5.6c illustrates the relative differences in piston velocity

with respect to time for this actuator. There is a relatively large, but short lived, spike at the discontinuity where the actuator encounters the spring-load; otherwise the differences are very small. In general there is some evidence of wave propagation effects (fluid inertia), which explains the transient disparity between the lumped and distributed parameter results.

§5.4.2.2 Performance Comparison

The most significant difference between the two methods is in the simulation time. The lumped parameter simulation, running on a Sun 4/370 Sparc processor, required 4860 seconds of CPU time. This was numerically highly stiff and discontinuous in parts of the simulation. With the usual relative error tolerance of 10^{-5} the LSODA computation required time-steps in the range $3.16\text{x}10^{-16}$ to $2.60\text{x}10^{-2}$ seconds and some 561 Jacobian re-evaluations. A total of 71341 Adams steps (orders 1-7) and 941 Gear steps (orders 1-5) were used and a substantial 20416 discontinuities processed.

The equivalent TLM solution required only 189 seconds of CPU time for comparable results, with time-steps in the range $1\text{x}10^{-6}$ to $1.84\text{x}10^{-3}$ seconds. The TLM algorithm is an explicit solver with very much fewer computational overheads than LSODA, thus a direct comparison of time step between the two methods as the solution proceeds may be misleading; a better measure is the recorded computation time against simulation time. The CPU times for both solvers as the solution proceeds are shown in Table 5.3. This data clearly demonstrates that the TLM solver consistently outperforms LSODA in this example, albeit with considerable variation in performance. The algorithmic "speed-up" shown in Table 5.3 is defined as the ratio of LSODA run-time to TLM run-time and is a measure of how much faster (or slower) the TLM solver is compared with the LSODA computation. The most significant TLM speed-up occurs when LSODA numerically solves a highly stiff system of ODE's, in this case roughly between simulation times 1.25 and 1.75s. TLM also requires much smaller time-steps to reduce the pressure error due to inductance in the transmission lines, but the magnitude of the steps selected are significantly greater than the order 10^{-16}s steps employed by the LSODA integrator. The two-actuator lumped-parameter simulation changes its numerical characteristics immensely during the simulation, from switching from stiff to non-stiff and back again. The different speed-ups shown in Table 5.3 gives an indication of the likely performance of TLM when used to solve hydraulic systems that are stiff or non-stiff when expressed as entirely lumped-parameter systems. This example indicates algorithmic speed-ups ranging from approximately

3 to 30 times, the greater the speed-up the stiffer the numerical integration (at these times LSODA generally used first order implicit numerical integration and very small time-steps).

Section 6.5.2 describes the TLM simulation of a hydraulic "ring-main" circuit, which has also been computed in parallel. For this example the TLM algorithmic speed-up compared with the lumped-parameter computation ranges from approximately 7 to 20 times faster throughout the simulation. Overall the ring-main lumped-parameter simulation is less stiff numerically and the ODE solution is computed faster in comparison with the TLM solution, hence the overall speed-up is not as good as the two-actuator example. However, the speed increases achieved using TLM for the non-stiff parts of both example simulations are of the same order of magnitude, ie. about 3 for the two-actuator circuit example, and 7 for the ring-main circuit example. This is also the case for the stiff parts of the example circuits, ie. a maximum algorithmic speed-up of about 30 for the two-actuator circuit and about 20 for the ring-main circuit. Thus, in practice the actual performance of the TLM computation versus the lumped-parameter computation will be very much problem dependent. Nevertheless, there is an improvement in simulation speed using TLM, which is likely to be even more significant when the lumped-parameter computation requires the numerical solution of a stiff system of ODE's.

Figure 5.7 (Table 5.2) shows the variation in TLM execution time with different values of the pressure error parameter and Figure 5.8 demonstrates how the pressure error tolerance used can influence the simulated response; in this case the inlet pressure to the pilot operated unloading valve is considered. Notwithstanding the variation in the predicted dynamics, the steady-state levels are still predicted correctly. However, transmission-line pressure error tolerances in excess of about 5 bar introduce significant distortion due to fluid inertia and are no longer comparable with the lumped parameter system. (The real system will include some fluid inertia, and in the absence of fully validated experimental data the lumped parameter model may be no more accurate in an absolute sense than the TLM solution; it does however give a useful guide to overall performance).

Figure 5.9 and Table 5.4 show the variation in run-time for fixed time step TLM simulations. In order to achieve results that correspond closely with the variable step method (ie. to locate discontinuity points with sufficient accuracy and minimise wave effects), a fixed time step of the order of 10μs is required. A 10μs simulation requires some 624s processor time. Although fixed time step TLM simulations can be very rapid, it is necessary to repeat simulations with progressively

reduced time steps to achieve confidence in the results obtained. With sensible values of transmission-line pressure error in the variable time step scheme, simulations are both very fast and have built-in error control. With a tolerance of 0.1 bar for example, the TLM simulation is approximately 25 times faster overall than the lumped parameter equivalent. Even without the use of parallel processing, remarkable speed-ups are possible using this distributed parameter modelling technique. The main reason why the lumped parameter LSODA integration is so slow in this instance is because the system is extremely stiff numerically when the pilot operated relief valve unloads the larger capacity pump (roughly between 1.25 and 1.75s).

It is interesting to measure the change in the performance of the LSODA simulation, with changes in global integration error tolerance. Table 5.5 shows the run-times obtained for different tolerances. In this example the fastest LSODA simulation was obtained with an increase in tolerance to 10^{-3}, although the transients had changed significantly from those computed using the recommended tolerance of 10^{-5}. Comparison of results between the 10^{-5} and 10^{-7} tolerance simulations showed no apparent differences. Experience with the LSODA integrator in the Fluid Power Centre suggests that the relationship between tolerance, numerical accuracy and speed is highly problem dependent. Paradoxically, a larger tolerance can sometimes cause an increase in run time (this is probably due to the propagation of numerical errors as the simulation proceeds making time step selection more erratic).

§5.5 Closure

In this chapter the process of automatic system model generation has been described for the case of single processor TLM simulations. The two-actuator test case example has been compared with an equivalent lumped parameter simulation, using the ODE solver LSODA. Overall, the TLM simulated transient results were very close to the lumped parameter solution, even during rapid transients. The steady-states results were indistinguishable.

The most remarkable difference, however, was noted in the computational performance of the TLM simulation employing the variable time step scheme. For the two-actuator example TLM demonstrated an overall speed-up in excess of twenty-five times the equivalent lumped parameter simulation; the speed-up varied by an order of magnitude as the simulation proceeded, from roughly 3 to 30 times faster depending on the stiffness of the lumped-paramter solution.

These results have been achieved using a single processor only; suitably partitioned TLM simulations, with separate sub-circuit elements running on different processors, should enable even greater speed-ups. This topic is the subject of the following chapter.

act1/act2:	fd:	cv1:	cv2:
F_S 100 [N]	D_{m1} 100 [cc/rev]	P_c 10 [bar]	P_c .5 [bar]
F_R 50 [N]	c_{f1} 0.1	K 100 [l/min/bar]	K 100[l/min/bar]
d_p 125 [mm]	c_{s1} $3x10^{-8}$		
d_r 70 [mm]	c_{v1} $2x10^{5}$	lines:	dcv:
M_e 10 [Kg]	D_{m2} 100 [cc/rev]	B_e 8900 [bar]	ΔP_{ref} 205 [bar]
c 3000 [N/(m/s)]	c_{f2} 0.231	P_o 0 [bar]	Q_{ref} 5 [bar]
x_o 1600 [mm]	c_{s2} $5x10^{-8}$	d 25 [mm]	
v_o 0 [m/s]	c_{v2} $3x10^{5}$	$L_1+L_3+L_7$ 4 [m]	u(t):
x_K 1700 [mm]	J 0.1 [Kg m^2]	$L_2+L_4+L_5+L_8+L_9$ 4 [m]	0 [] [t < 0s]
K 1.8 [KN/mm]	c 0.1 [Nm/rpm]	$L_{10}+L_{11}+L_{12}$ 3 [m]	1 [] [t=0-2.5s]
s 1850 [mm]		$L_{13}+L_{14}+L_{15}+L_{17}$ $+L_{18}$ 4 [m]	
	rv1/rv2/rv3:	$L_{19}+L_{25}+L_{28}$ 8 [m]	
p1:	P_c 210 [bar]	$L_{20}+L_{26}+L_{27}$ 8 [m]	$f_1(t)/f_2(t)$:
Q_p 121 [l/min]	K 600 [(l/min)/bar]	$L_{21}+L_{23}+L_{24}$ 1 [m]	0 [N] [t=0-2.5s]
		$L_{22}+L_{29}+L_{30}$ 12 [m]	
p2:	system parameters:		prv:
Q_p 84 [l/min]	ρ 890 [Kg/m^3]	tk:	P_c 145 [bar]
	ν $60x10^{-6}$ [m^2/s]	P_t 0 [bar]	K 600[l/min/bar]

TABLE 5.1 TWO-ACTUATOR CIRCUIT PARAMETRIC DATA

Pressure *error* **Pe** [bar]	Simulation Run-time [s]
0.01	471.4
0.1	189.0
0.25	49.9
0.5	24.3
1.0	13.0
2.5	7.2
5.0	2.6

TABLE 5.2 VARIABLE-STEP TLM RUN-TIME

Simulated Time [s]	LSODA CPU Time [s]	TLM CPU Time [s]	Speed-up [s]
0.25	-	-	-
0.5	250 (250)	17 (17)	15
0.75	-	-	-
1.0	281 (31)	18 (1)	31
1.25	311 (30)	20 (2)	15
1.5	3450	141 (121)	26
1.75	(3139)	160 (21)	30
2.0	4849	188 (1)	8
2.25	(1399)	-	-
2.5	4857 (8)	189 (1)	3
	-		
	4860 (3)		

TABLE 5.3 TLM/LSODA ALGORITHMIC PERFORMANCE

[Note: figures in brackets refer to processing times needed to compute the time interval]

Fixed Time-step Δ [μs]	Simulation Run-time [s]
10	624.4
100	62.8
250	23.6
500	11.7
1000	6.3
2500	2.4

TABLE 5.4 FIXED-STEP TLM RUN-TIME

LSODA tolerance	Simulation run-time [s]
10^{-1}	13136.4
10^{-3}	3329.9
10^{-5}	4860.0
10^{-7}	9185.0

TABLE 5.5 LSODA RUN-TIME

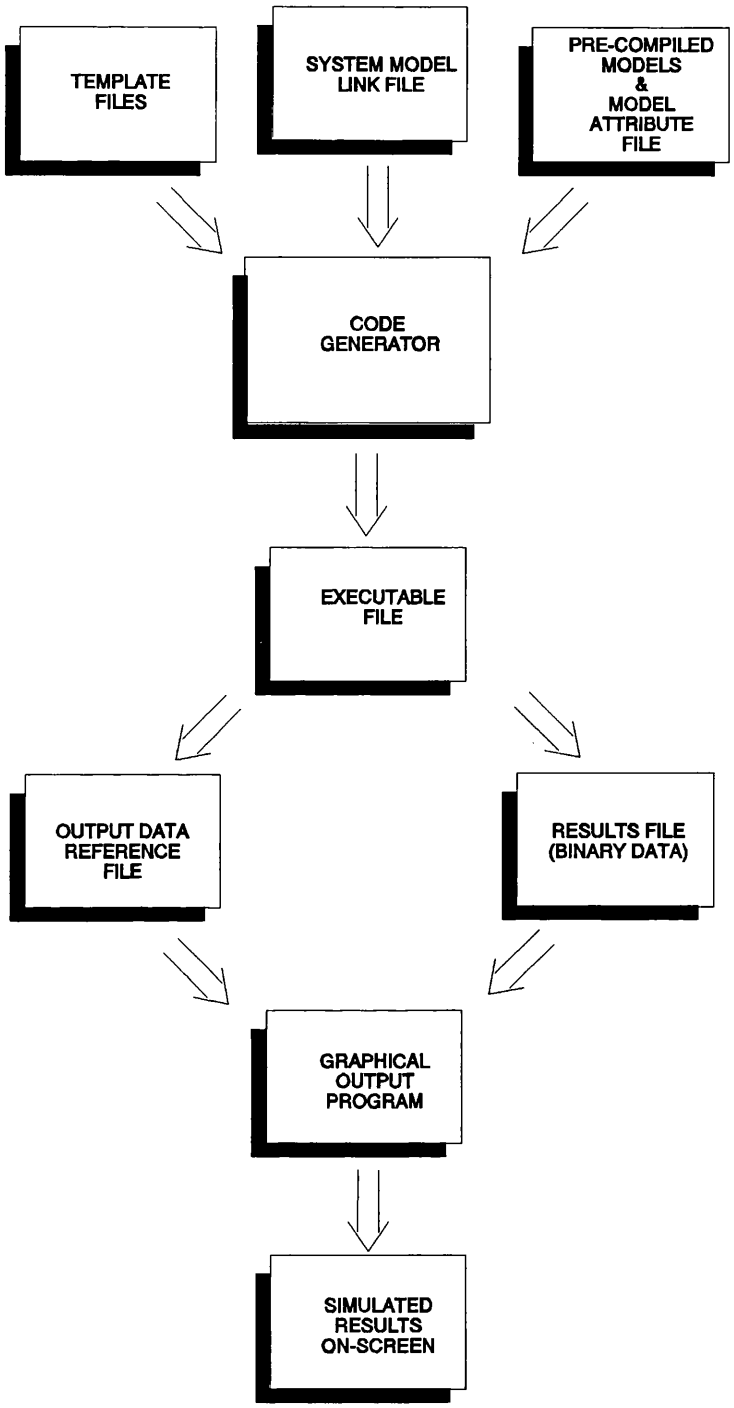


FIGURE 5.1 SIMULATION PROCESS: AUTOMATIC CODE GENERATION

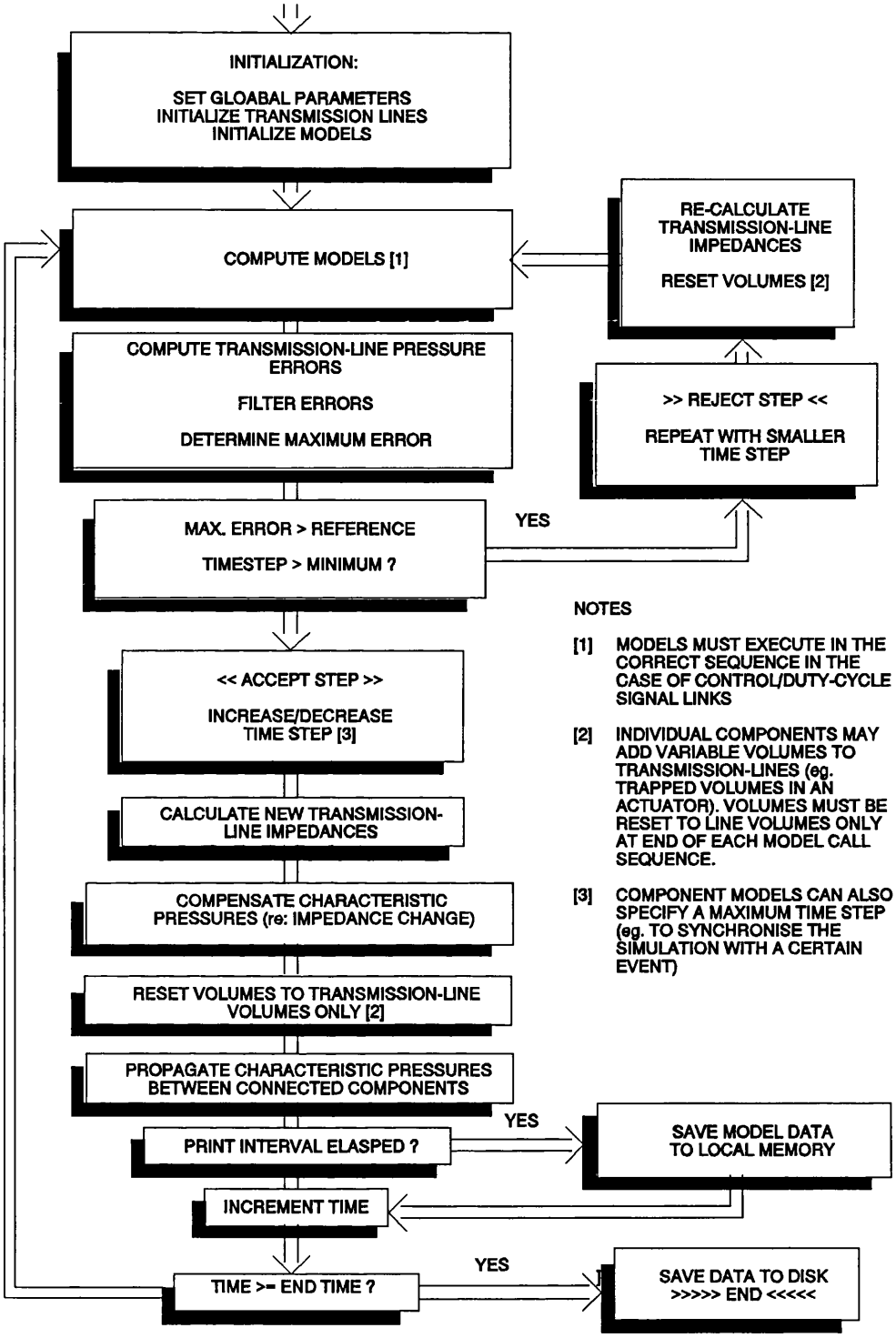


FIGURE 5.2 TLM ALGORITHM

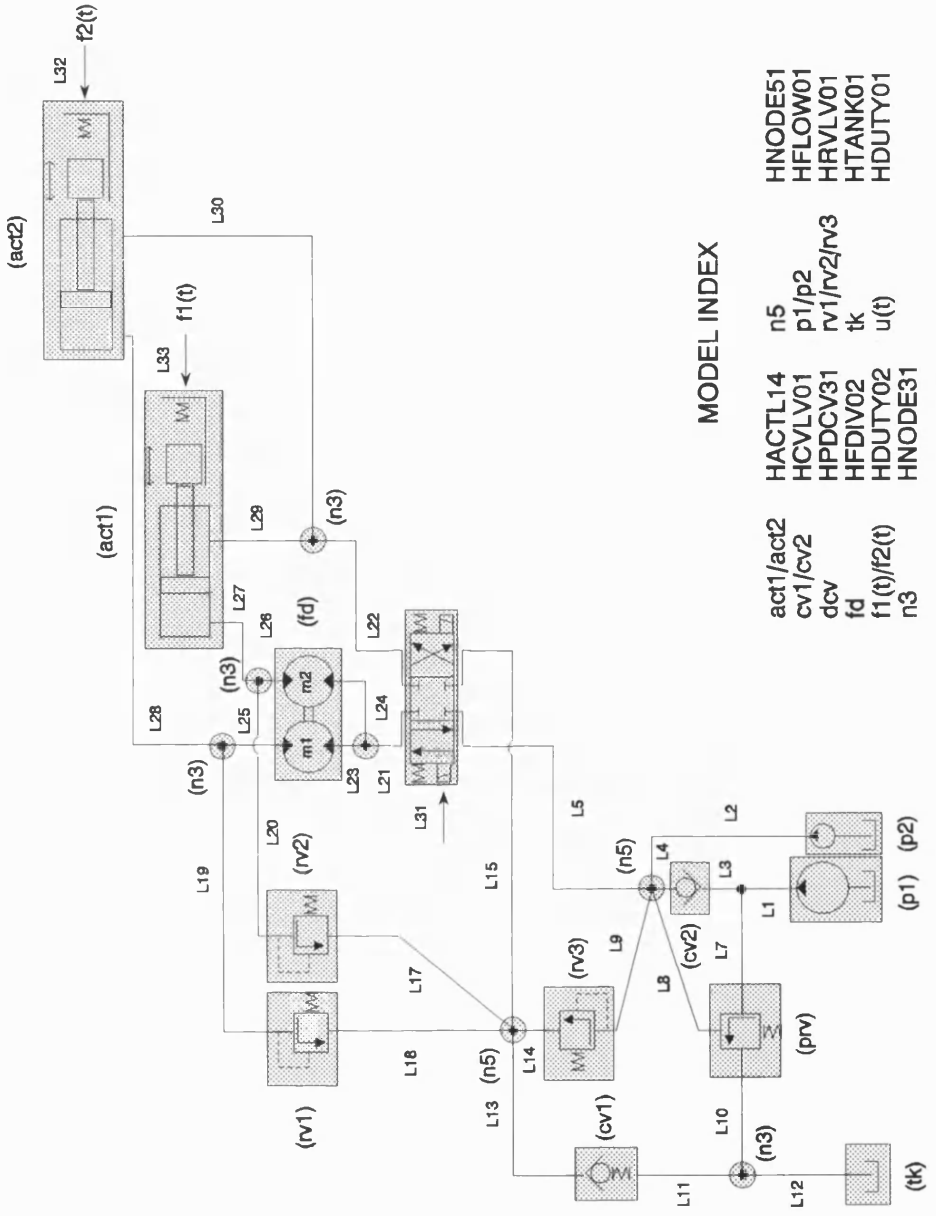


FIGURE 5.3 ACTUATOR CIRCUIT MODEL LINK DIAGRAM

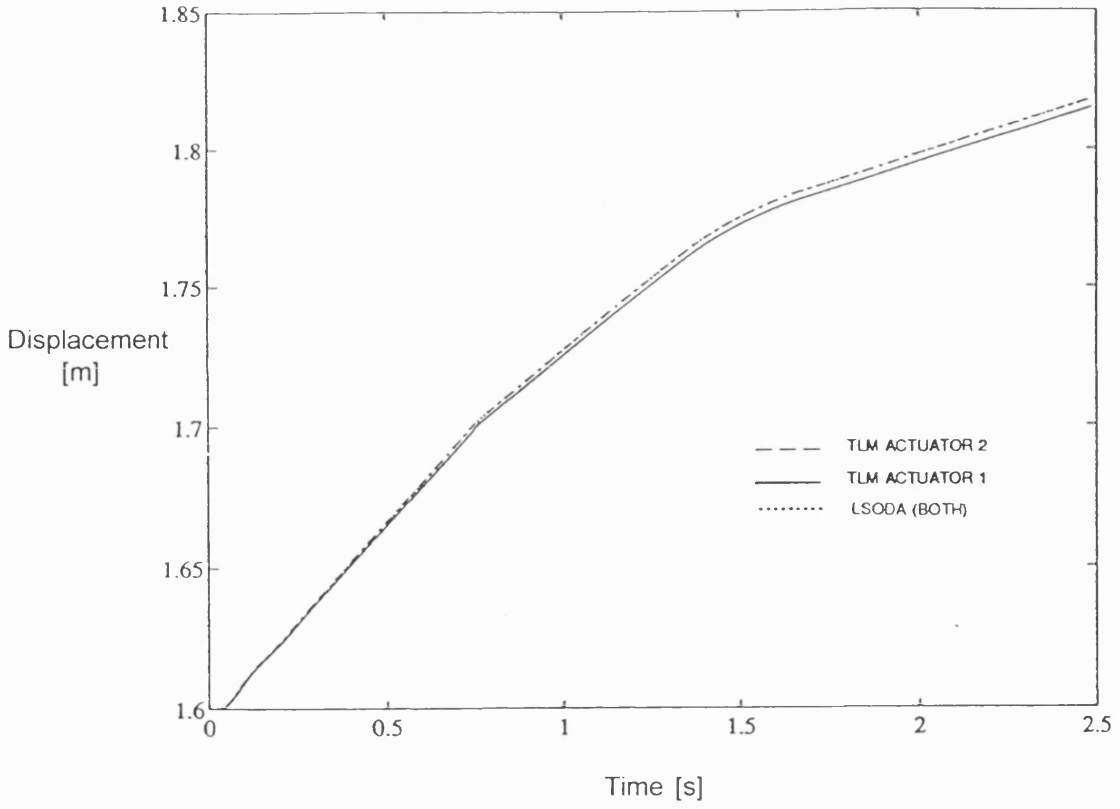


FIGURE 5.4a ACTUATOR 1 & 2 DISPLACEMENT TRANSIENTS

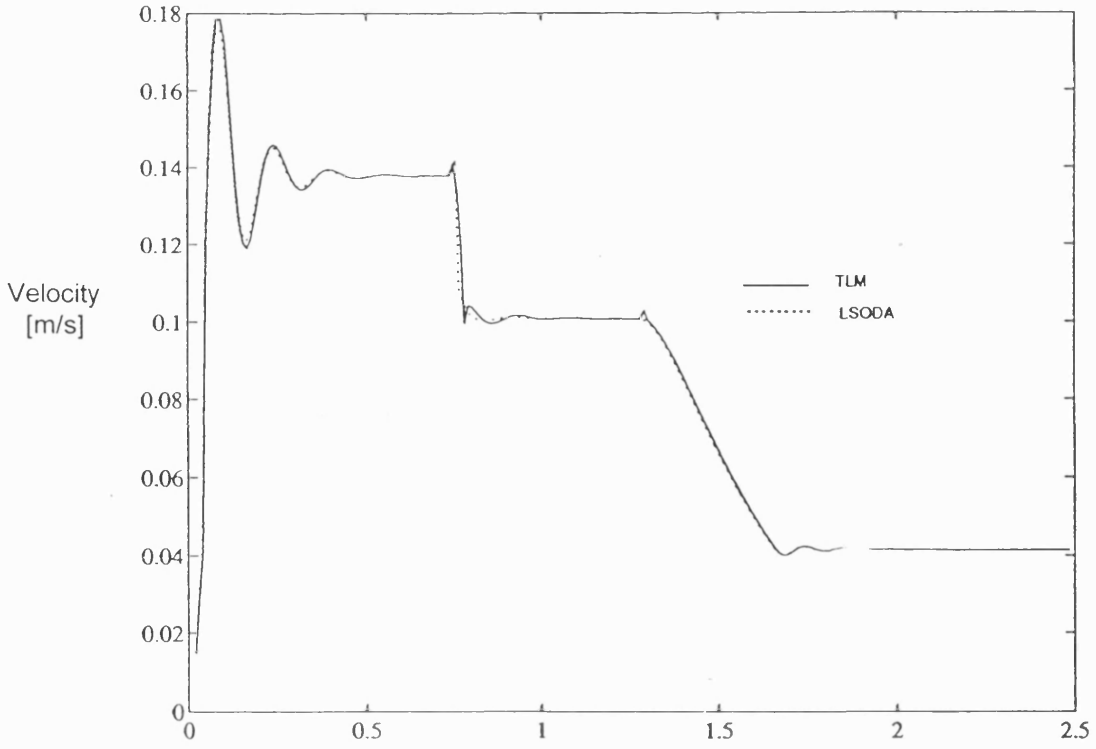


FIGURE 5.4b ACTUATOR 1 VELOCITY TRANSIENT

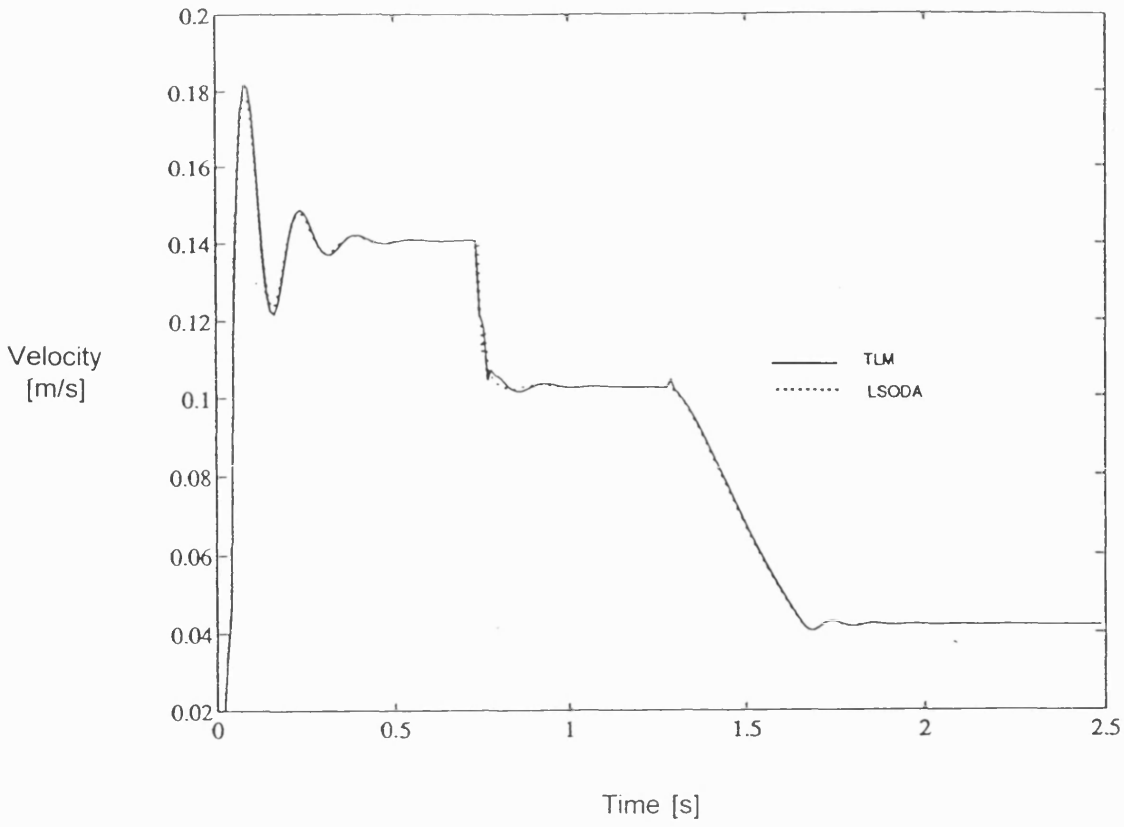


FIGURE 5.4c ACTUATOR 2 VELOCITY TRANSIENT

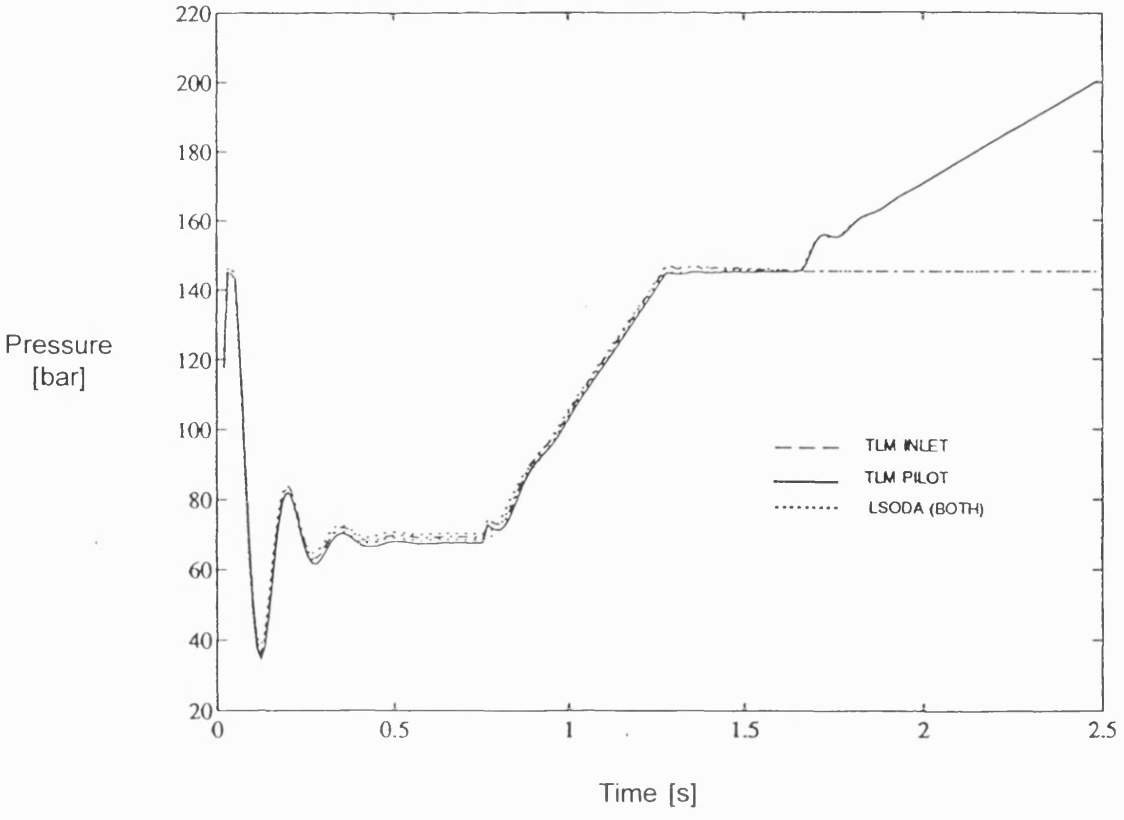


FIGURE 5.4d UNLOADING VALVE INLET & PILOT PRESSURE TRANSIENTS

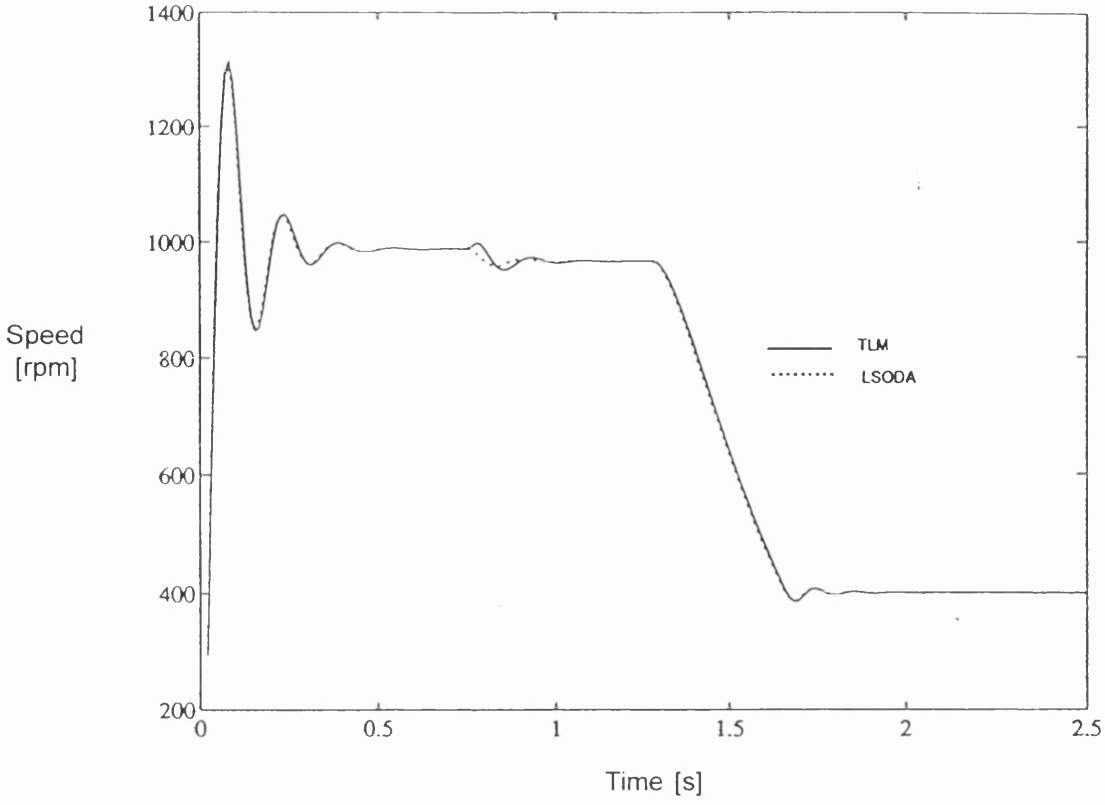


FIGURE 5.4e FLOW-DIVIDER SHAFT SPEED TRANSIENT

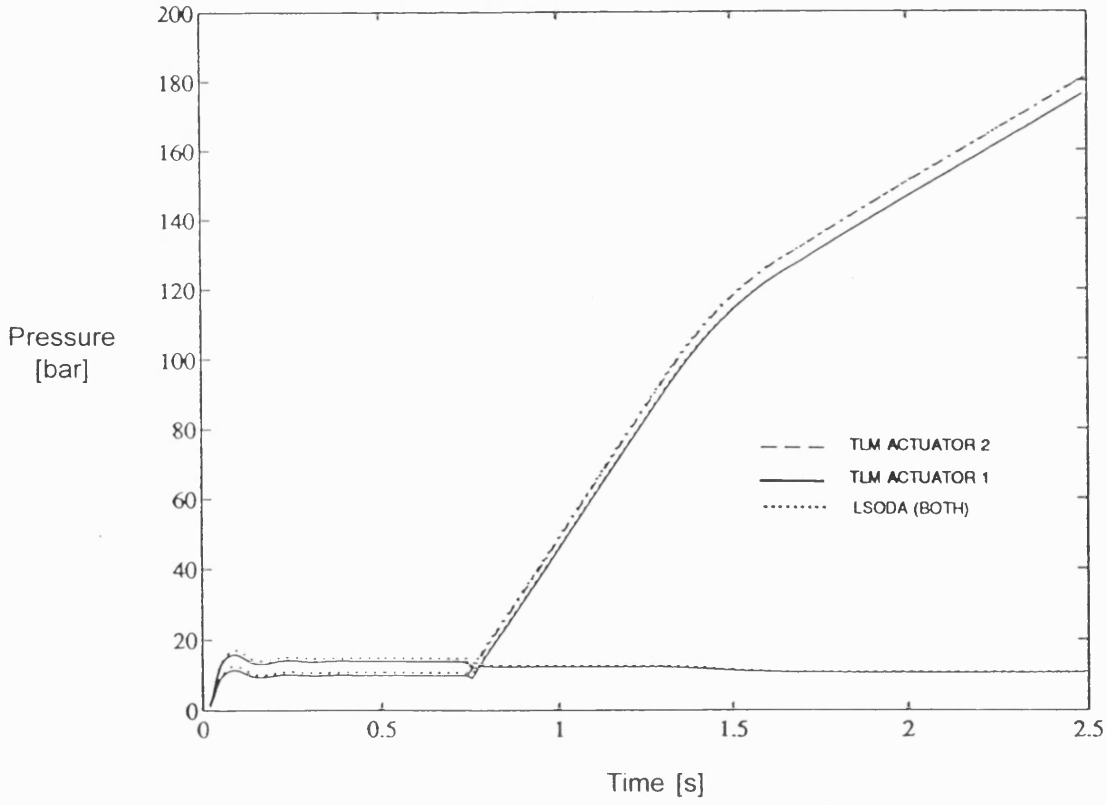


FIGURE 5.4f ACTUATOR 1 & 2 PISTON-END AND ROD-END PRESSURE TRANSIENTS

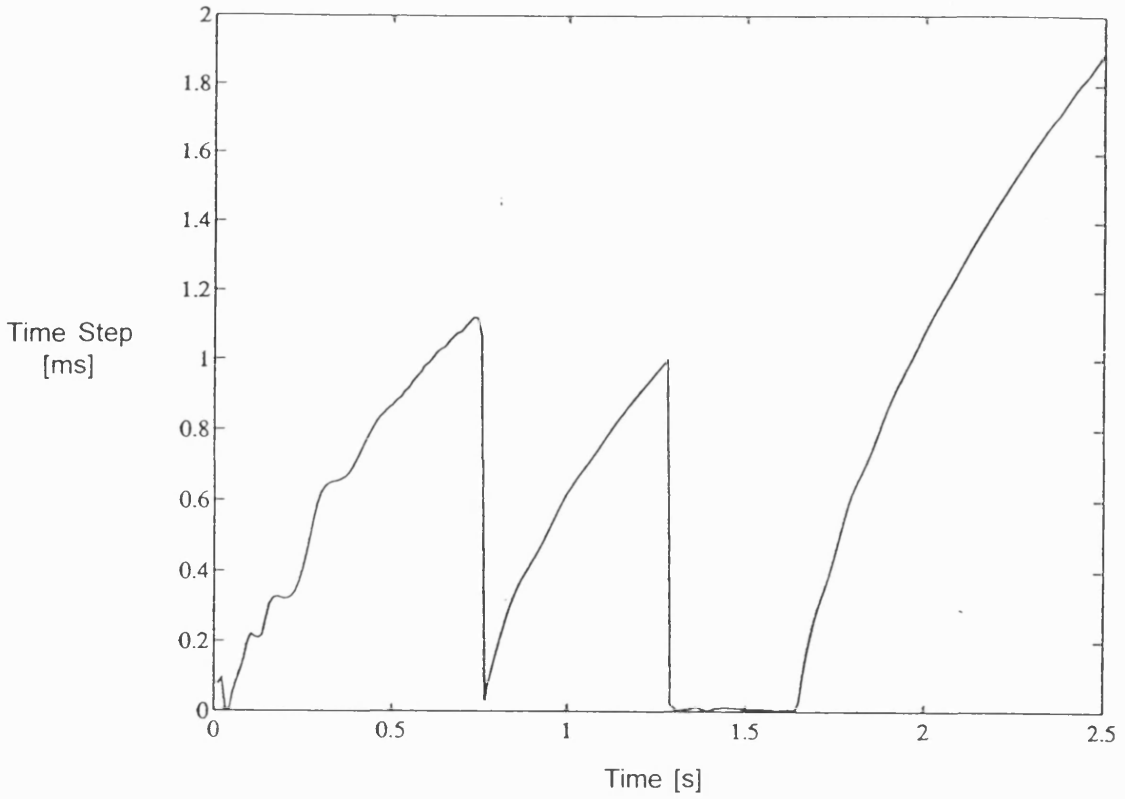


FIGURE 5.5a TLM TIME STEP [Pe 0.1 bar]

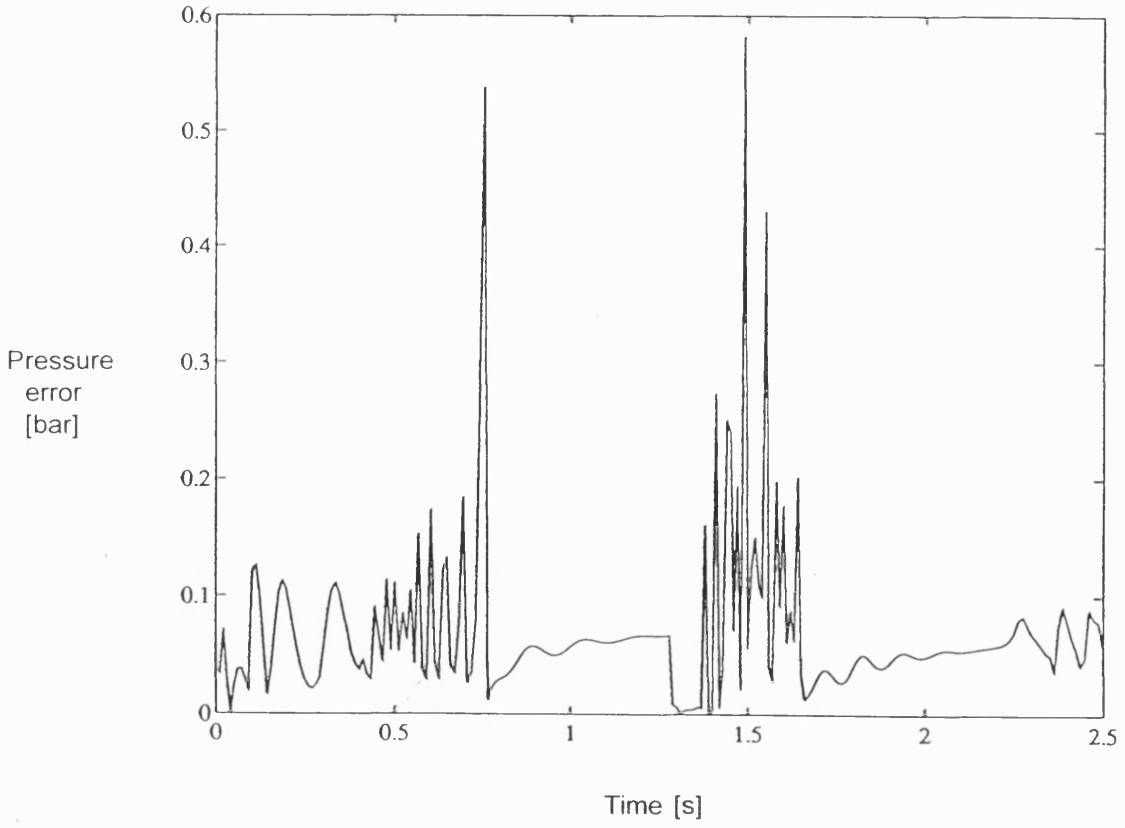


FIGURE 5.5b TLM MAXIMUM TRANSMISSION-LINE PRESSURE DIFFERENCE [Pe 0.1bar]

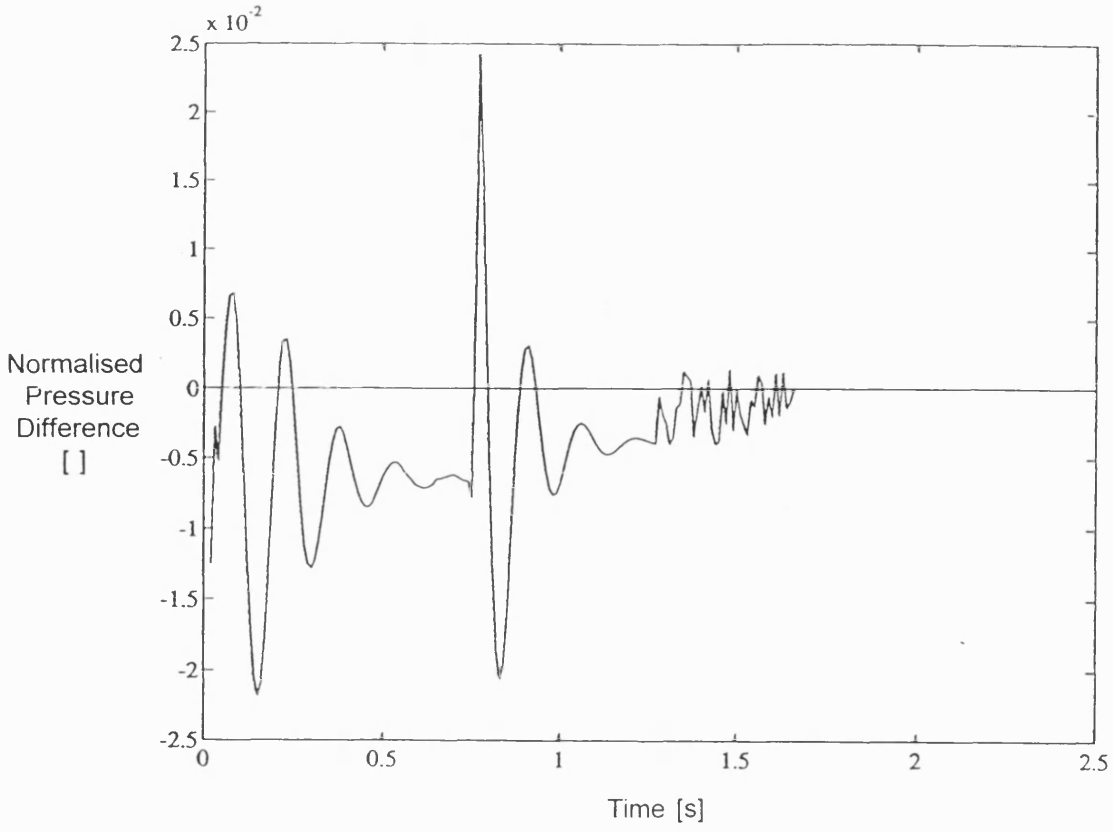


FIGURE 5.6a TLM-ODE RELATIVE DIFFERENCE TRANSIENT: UNLOADING VALVE INLET PRESSURE

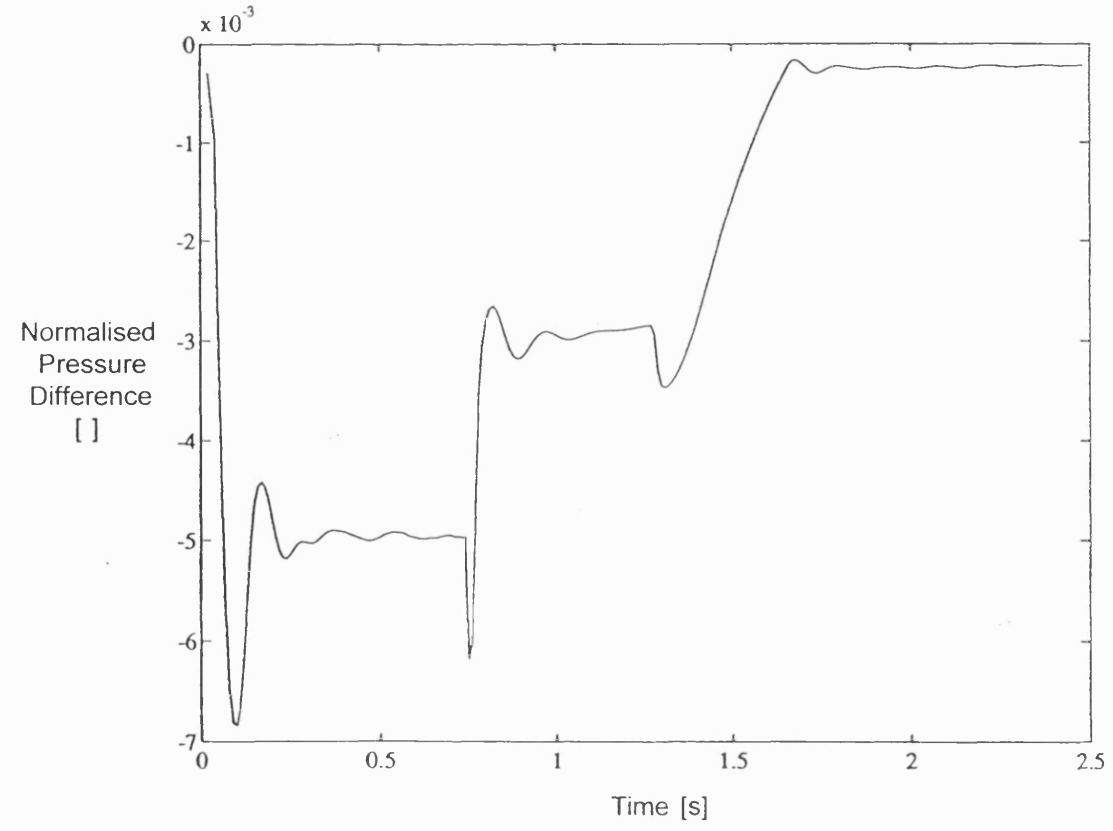


FIGURE 5.6b TLM-ODE RELATIVE DIFFERENCE TRANSIENT: ACTUATOR 1 PISTON-END PRESSURE

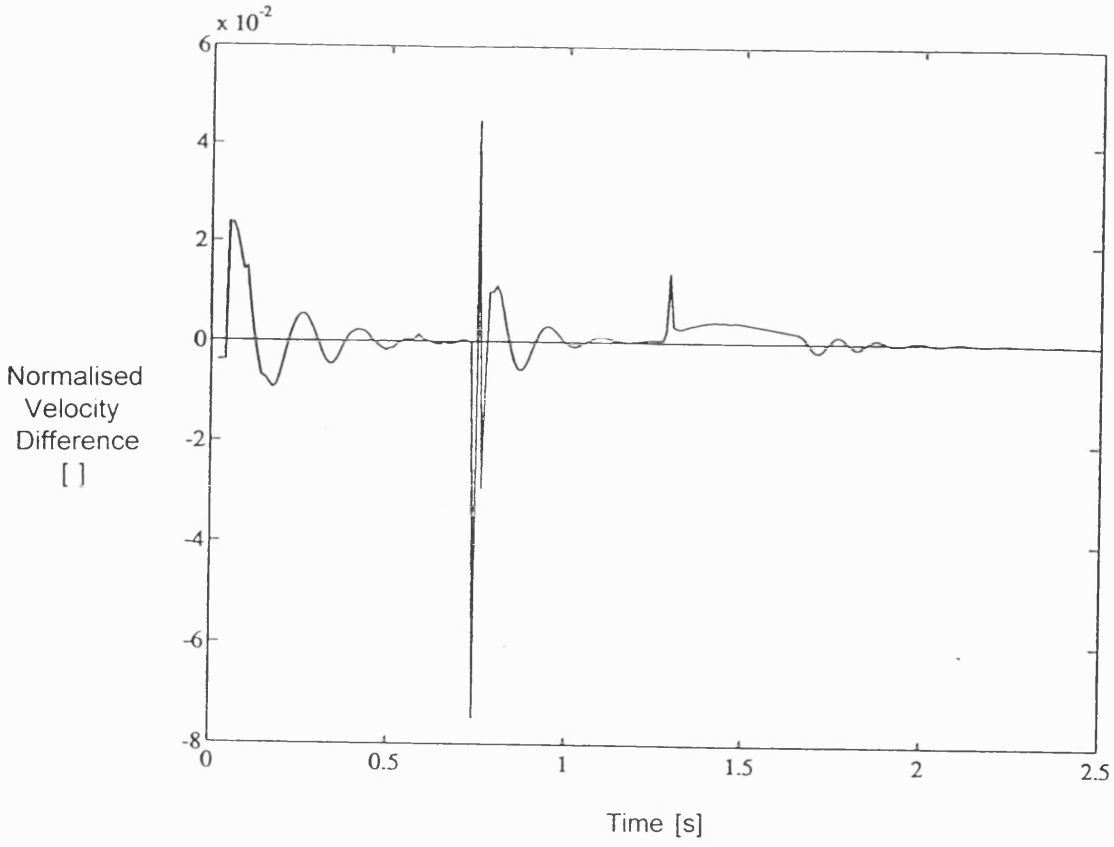


FIGURE 5.6c TLM-ODE RELATIVE DIFFERENCE TRANSIENT: ACTUATOR 1 VELOCITY

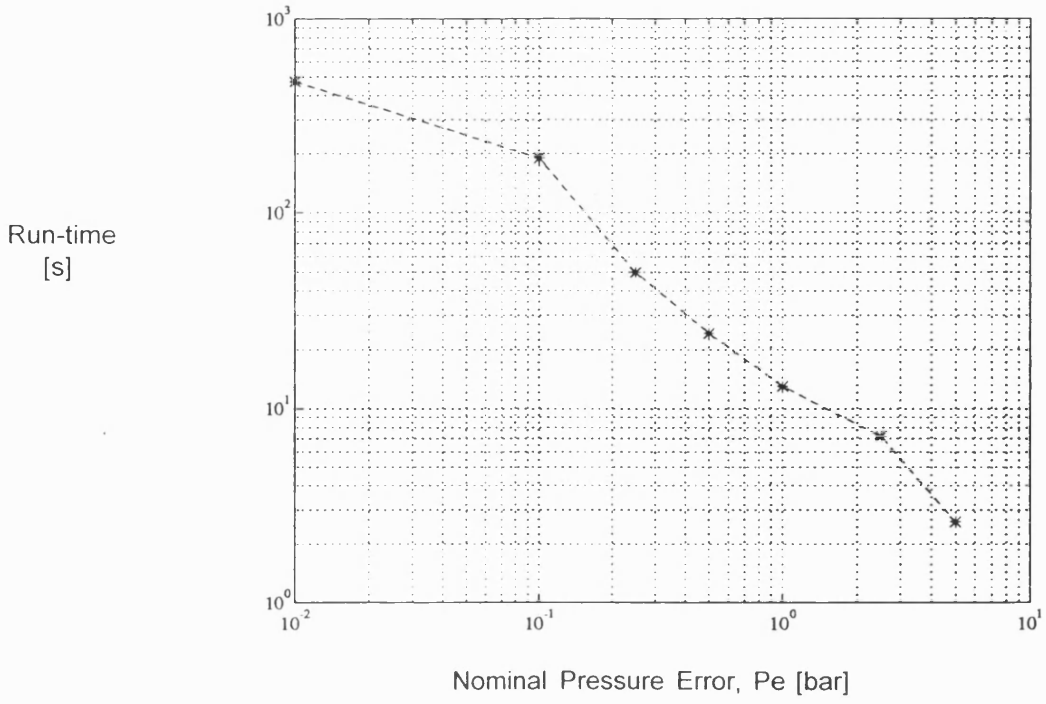


FIGURE 5.7 RUN-TIME AGAINST TLM PRESSURE ERROR: VARIABLE-STEP TLM

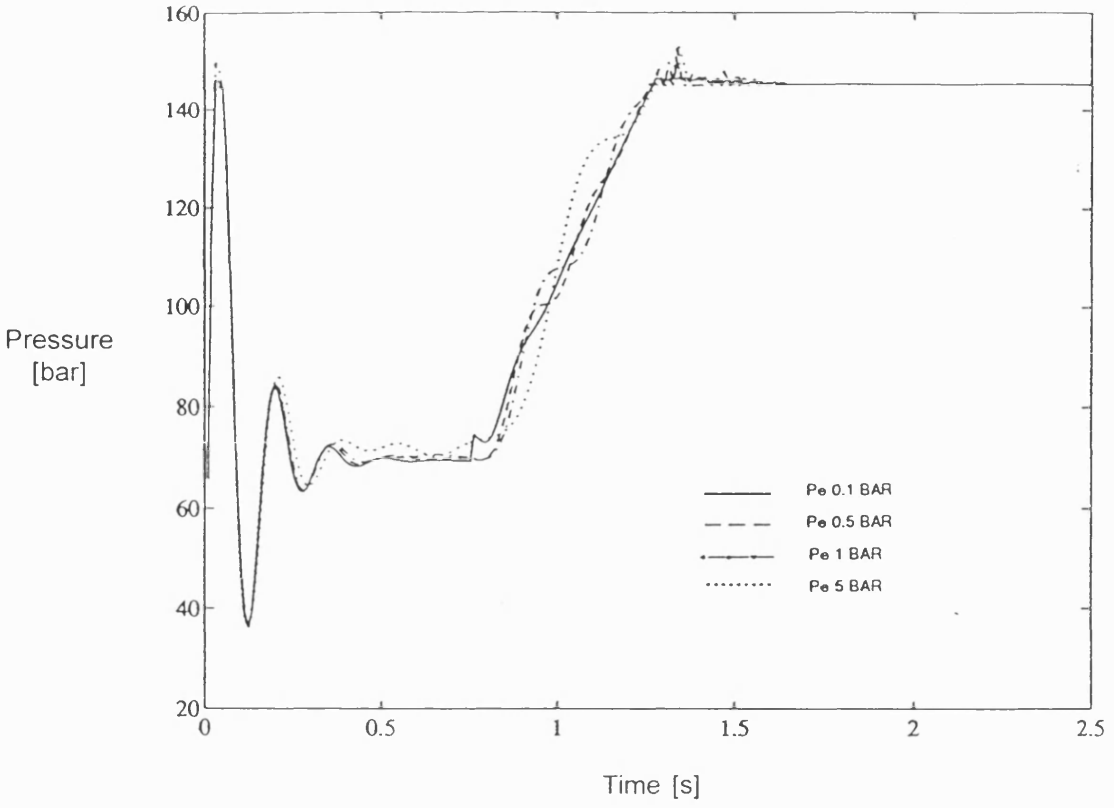


FIGURE 5.8 UNLOADING VALVE INLET PRESSURE TRANSIENT: VARIATION WITH TLM PRESSURE ERROR, Pe

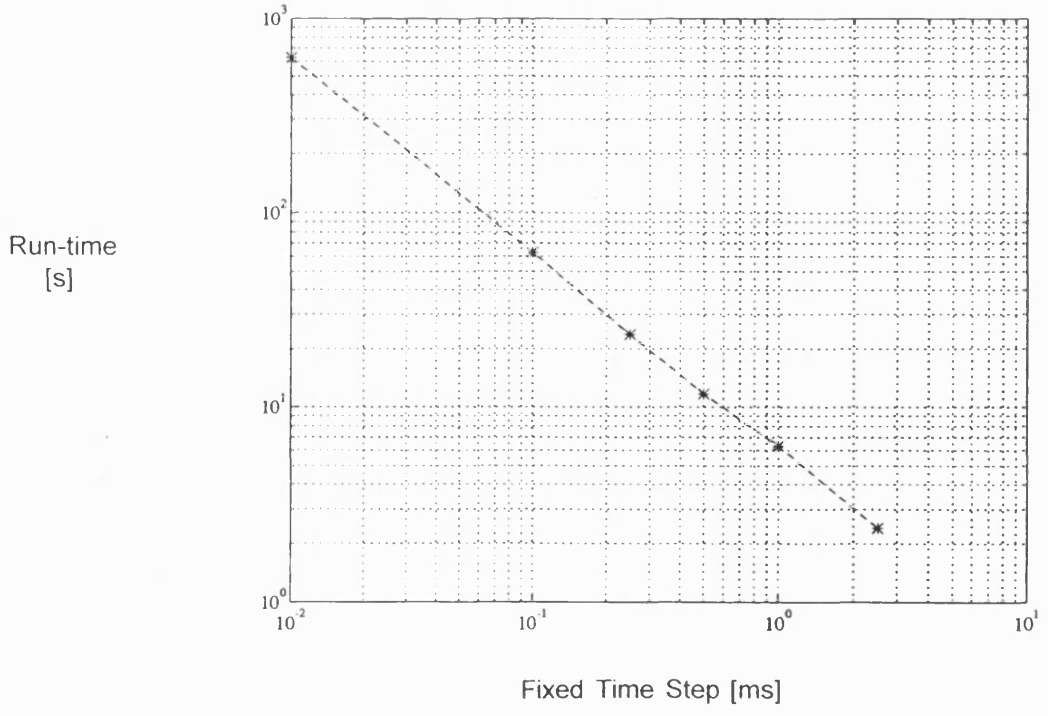


FIGURE 5.9 RUN-TIME AGAINST TIME STEP: FIXED-STEP TLM

CHAPTER 6

PARTITIONED SIMULATION

---

§6.1 Introduction

The two-actuator example presented in Chapter 5 has provided a simple demonstration of the computational speed increase that can be achieved by representing a hydraulic system using transmission-line modelling (TLM). A numerical solver has been demonstrated which uses TLM to distribute the solution algorithm throughout the problem domain, into the separate component models, using a single processor. Conceptually, reconfiguration of the TLM scheme onto a parallel computing surface is straight-forward, as the transfer of delayed characteristic pressures between component models either takes place locally on the same processor via "on-chip" memory, or with connected processors via serial data links.

§6.2 Partitioning

§6.2.1 Domain Decomposition

The highest level of decomposition that can be applied to a circuit is to place each component onto its own processor. However, this is generally undesirable due to the communications overheads that may be imposed. It is preferable to group components together on a processor; each group of components representing a subsystem of the circuit; this approach is known as *domain decomposition.* Each pipe between subsystems therefore represents an information transfer between domains, each domain computed on a separate processor. Burton *et al* [1993B] discuss this approach applied to the two-actuator example of Chapter 5 for different numbers of processors and model-to-processor partitioning arrangements, using fixed time-step simulation.

§6.2.2 Simulation Control using the Master-Slave Configuration

If a variable time-step method is employed then an additional requirement is placed on partitioning; only one processor can be allowed to determine the time-step. Consequently the simulation must be partitioned such that one subsystem exchanges information with all others. The master processor governs the operation of the slaves, by defining the simulation time-step used in the computation. The value of time-step calculated by the master process is determined from error control information obtained from both the master and the slave partitions.

The use of a master, or central control, process means that the predictions obtained will be identical to those computed by an equivalent single processor simulation. This approach differs from the parallel scheme first used by Jansson *et al* [1992], in which each partition has its own individual step controller. Synchronisation between the sub-systems is problematic in Jansson's approach (owing to the disparate time steps), resulting in time step oscillations as the separate controllers interact at the communication time steps. Moreover, the simulation accuracy is reduced as the step rejection procedure must be disabled for this scheme to operate (if a step is rejected it must be rejected for all partitions).

The master-slave configuration overcomes these difficulties although quiescent, or slow dynamic, parts of a circuit are subjected to the same (potentially) small time steps needed elsewhere in the circuit model. This limitation does not affect adversely multi-processor efficiency, provided the individual processor loads are balanced evenly throughout the distributed computation. The separate controller technique will not be measurably faster, because of the unequal computational loads evident when one partition requires a very small time step compared with the others; multi-processor speed is always limited to the speed of the slowest partition, as with the master-slave scheme.

§6.2.3 Parallel Efficiency

The performance of parallel computations is governed largely by the ratio of compute-time to communications-time for well-partitioned systems. For this reason full multi-processor operation (ie. one processor per model) is unlikely to be an ideal (or possible) configuration. However, the process of problem decomposition can often be very adaptable, as different models can be assigned to separate processors in many different combinations. To improve efficiency, connected component models should be partitioned together into sub-circuit *topological groups*, reducing the number of

inefficient inter-processor data transfers. Moreover, balanced computational loads are necessary, if parallel simulations are to increase simulation speed with the addition of further processors; the maximum attainable speed will be dictated by the slowest partition, ie. the partition with the most computations per step.

"Speed-up" is the term that refers to the increase in execution speed for a multi-processor computation, compared with the equivalent single processor computation. The "ideal" speed-up is equal to the number of processing elements, which implies exact load balancing and zero communications delay. A measure of parallel efficiency is therefore obtained from the ratio of the actual speed-up achieved to the number of processors used.

The optimum partitioning arrangement is very much dependent upon the increased parallelisation afforded by the use of more processors, the distribution of computational loads across these elements and the penalty of communications between them. Consequently, efficient operation is highly problem dependent. Moreover, some models may change their computational requirements during program execution, as different operating regions are encountered, upsetting the balance of processor loads. This point is addressed further in subsequent simulation examples.

§6.3 Partitioned Variable-Step TLM Algorithm

The nucleus of partitioned TLM is the parallel variable-step algorithm. This numerical scheme relies on a designated *master* to direct the process of error control and time step calculation for the complete distributed simulation. Both master and slave processes compute separate sub-circuit partitions concurrently. Figure 6.2 shows this scheme diagrammatically for a two-processor master-slave system.

At each step, following the execution of each component model, the maximum transmission-line pressure error in each slave is transmitted to the master together with the characteristics and pressures of all connecting transmission-line links. From the maximum overall error the controller then either accepts, or rejects the computed step and calculates the magnitude of the next step. This procedure automatically allows for the effect of any discontinuity. Step control (an accept/reject flag plus the new time-step) and characteristic pressure inputs are then transmitted to the slaves. This cycle is repeated until the simulation is complete.

In summary, the simulation process is distributed throughout the component models, in the separate

sub-system partitions. Multi-step control is facilitated centrally by use of a step controller (the master process). It is this control process that maintains the synchronisation between the individual partitions.

§6.4 Distributed Processing

§6.4.1 Hardware Description

The parallel computing surface used for this research comprised a Sun 4/370 host computer, supporting a Transtech MCP1000 transputer motherboard, itself containing a total of eight processing nodes, or "TRAMs". Each TRAM (an acronym for transputer plus random access memory) consisted of an Inmos T800 transputer (a 25MHz CPU, cache RAM and four high speed serial data links) attached to 2Mbytes of local memory, depicted in Figure 6.3. The schematic layout of the MCP1000 is shown in Figure 6.4 and illustrates the possible connectivity of the multi-processor system. There are two TRAMs per site (out of a possible four), both of which are permanently wired together. The remaining transputer links may be inter-connected via the link switch adaptor, to configure a wide range of processor topologies, except for link 1, slot 1, which facilitates a connection to the host computer (file and screen access, for example). The remaining *free* transputer links can be *software configured* from the host by appropriate use of the Transtech "**nt_ctl**" utility. This configuration software is supplied as standard with the Transtech MCP1000 transputer motherboard and described fully in the Transtech user manuals.

§6.4.2 Software Description

Initially, the transputer operating system "Genesys" (© Transtech Devices) was used, consisting of the operating system itself (a Unix-like system for transputers), in addition to the associated libraries, "include" files and the Genesys C compiler, to enable message passing between processes on the same and on separate processors. Preliminary investigations demonstrated conclusively that the computational overheads associated with the Genesys system were very significant, increasing the time required to send and receive messages between processors to an unacceptable level, even resulting in multi-processor "speed-downs". A considerable amount of time and effort was expended in setting up the Genesys simulations, only to discover that this approach was not a viable one. A major problem was the lack of expertise and support from the vendors (Transtech) and the

experimental nature of the Genesys transputer operating system.

Consequently, the Inmos "ANSI C ToolKit" and "Virtual Route Configurer" were applied to generate the application-specific transputer object code used in the multi-processor examples discussed later. This software enables executable code to be created without the need for a dedicated transputer operating system; an executable "image" (operating system-like instructions plus application code) is loaded onto the transputer system directly. The parallel software also included some additional multiplexing algorithms to enable every transputer to access the host computer's visual display (screen) and hard disk; this is essential in order to monitor the parallel simulation in progress and allow simulation results from each partition to be saved at the end of the run.

The Inmos T8 series transputer has a basic limitation of four bi-directional serial data links. For master-slave processor topologies this means that a five-processor configuration represents the greatest partitioning arrangement achievable using "nearest-neighbour" communications between processors. The alternative is message "through-routing", which involves sending data via an intermediate processor, or processors. This procedure is simplified (for the programmer) by the automatic multiplexing routines imbedded into the Inmos Virtual Route Configurer. Clearly, the consequence of failing to use nearest-neighbour data transfers is an increase in transmission delay and reduced efficiency. Figure 6.5 shows the range of possible processor topologies using both nearest neighbour and through-routed processor links.

§6.4.2.1 Program Development

The construction of transputer executable code involves the specification of the following files:

(i) Hardware and software configuration file, a "*filename*.**cfs**" script

(ii) Application code segment files for each sub-circuit partition

Each program segment (one per processor) must be compiled and linked using the Inmos ANSI **C** compiler and linker, "**icc**" and "**ilink**", respectively. Following compilation and linking, the program segments must be configured appropriately using an appropriate configuration file, which describes the connection between processors and processes. Subsequently, the Virtual Route Configurer, "**icvconf**", is invoked to produce executable code, in the form of a "*filename*.**btl**" file. This file is loaded on to the hardware via the interactive file server and loading program "**iserver**".

The above sequence of operations is standard for all parallel implementations using the Inmos ANSI

C Toolkit and Virtual Route Configurer and are described in detail in the user manuals supplied with the software. Example files and the sequence of operations necessary to construct parallel TLM code are given in *Burton 1994*, relating to the two-processor implementation of the circuit discussed in §6.5.1.

§6.4.2.2 Data Transfer

The standard form of data transfer between partitions (program segments) is uni-directional. The ANSI C Toolkit provides two functions to enable message passing between partitions, "**ChanOut**" (send) and "**ChanIn**" (receive) function. In order to construct a software data link between sub-systems over the hardware serial data link, one partition must contain a **ChanIn** and the other partition a corresponding **ChanOut** function call. Hence to achieve bi-directional data transfers between partitions, as required by the TLM algorithm, there must be two opposing uni-directional software links.

§6.4.2.3 Code Generation

The development of a complete multi-processor *code generator* is a complex task beyond the scope of the present investigation. However, the single-processor (host computer) *code generator* discussed in the last chapter, with few modifications, has enabled individual partitions to be generated automatically using separate sub-circuit *link* files; the multi-step controller (master process) and the inter-processor communication functions were then added manually. It was also necessary to provide an appropriate configuration file script to facilitate creation of an executable file, suitable for the processor topology established using the configuration tool "**nt_ctl**".

§6.5 Multi-Processor Applications

The following examples have been restructured for multi-processor simulation using the multi-step TLM algorithm developed. In order to test the applicability of this technique, two hydraulic circuits of differing sizes and complexity were investigated. The basic criteria was to investigate representative circuit models in order to test how well the parallel TLM algorithm scaled up (or down) between circuits containing widely different numbers of similar components. The small and large circuits studied are referred to subsequently as the "small-scale" and "large-scale" systems

respectively.

§6.5.1 Small-Scale System: Electro-Hydraulically Controlled Two-Actuator Circuit

Figure 6.6 shows an electro-hydraulic circuit containing two position-controlled actuators, each supplied by the same flow source (*Burton 1994* details the single processor circuit *link* file, including the physical data and component models used). Circuit functionality is similar in principle to the two-actuator example evaluated in Chapter 5, except that position control of the actuator pistons is governed electro-hydraulically via feedback control, as opposed to hydraulically by application of a flow-divider. This circuit combines instantaneous control elements with variable volumes and several discontinuities in the form of relief valves and actuator end-stops.

§6.5.1.1 Control Signals

Feedback control, as outlined in §3.7, is facilitated by the use of uni-directional *signal* links between models [Burton *et al*, 1993A][5]. Modelling electrical control signals requires component models in the feedback loop to be called in the correct sequence, following the direction of information transfer along the instantaneous signal link. In this example the correct sequence is: the feedback device (the transducer model imbedded into the actuator), the controller duty-cycle (input demand), the proportional controller and the modulating device (the 4-way, 3-position, closed-centre valve). This sequence is inserted into the simulation *link* file by the user. This process can be automated at a later date, as the connectivity of the feedback link is readily determined.

Other models with signal inputs and outputs may be included in the feedback loop, for example signal nodes, or summing junctions. The remaining component models may be executed in an arbitrary sequence, as they are decoupled completely by transmission-lines.

§6.5.1.2 Transient Response

Typical simulated transient responses of the system to a step input in actuator positional demand are shown in Figure 6.7, for both the LSODA (Gear/Adams) and the multi-step TLM methods; these simulations were performed on the host processor of the **Sun 4/370** computer, which in absolute

---

[5] Digital control can also be incorporated into the multi-step scheme, by forcing the step controller to adjust the time step to coincide with the sample interval

terms is approximately 35-40% faster than a single T800 transputer.

The use of multi-step TLM with $\mathbf{P_e}$=0.1 bar provides an overall algorithmic speed-up exceeding thirty times that achieved using the lumped parameter model for comparable results. The variation in processing time with simulation time for both solvers in shown in Table 6.1. This shows a variation in TLM speed-up from barely 3 to almost 50 times that achieved using LSODA.

The LSODA computation was numerically very stiff in parts of the simulation. For the (recommended) relative error tolerance of 10^{-5}, 112 Jacobian re-evaluations were required and 2150 discontinuities processed. Additional statistical information includes a minimum time-step of 3.162×10^{-16} seconds and a maximum time-step of 0.522 seconds, requiring 7027 explicit Adams integration steps (orders 1-7) and 520 implicit Gear integration steps (orders 1-5). TLM on the other hand is a single step explicit algorithm based upon wave propagation, which explains why TLM is faster even though it sometimes uses smaller time-steps (for this example the time-steps used were in the range 5.557×10^{-6} seconds to 2.5×10^{-3} seconds).

In order to obtain improved performance from the TLM solver, a multi-processor configuration must be employed.

§6.5.1.3 Partitioned Simulation

Component model call ordering, as required for cases involving control signals, imposes certain partitioning constraints, because the different sub-circuit partitions can only be inter-connected by transmission-lines. Consequently, in this case, the feedback loop consisting of actuator (transducer), duty-cycle (demand), controller and valve must be retained on the same processor to achieve correct signal propagation.

The different sub-circuit partitions studied and the corresponding processor topologies employed are illustrated in Figure 6.8 and the respective simulation speed-ups for each partitioning scheme detailed in Table 6.2, where speed-up is defined as the single processor run-time divided by the measured multi-processor run time. The speed-ups for a different input demand sequence (actuator positional demand) are also given and these show little change. In the latter case the positional demand (to both actuators) was changed to the fully retracted position (zero displacement) after two seconds of a four second simulation run. Figure 6.9 shows the measured and the ideal speed-ups for this system plotted against the number of processors employed. The corresponding parallel efficiencies are shown in

Figure 6.10, where this measure of efficiency is defined simply as the measured speed-up divided by the number of processors (the number of processors equals the maximum possible speed-up). The range of measured speed-ups and efficiencies corresponds to variations in partitioning and/or duty-cycle input to the parallel simulation. These results demonstrate that only a limited computational advantage can be obtained for parallel simulation of such a small-scale system. A maximum speed-up of 1.64[6] was obtained for the three processor simulation, using the T800 transputer. The equivalent three-processor T805 transputer simulation demonstrated a more impressive speed-up of 1.92 (64% efficiency), although no other configurations were possible using this transputer system.

Limited speed-up (reduced efficiency) is a consequence of problem "granularity", ie. the suitability of the T8 series transputer to the "fine-grained" parallelism problems encountered with the small-scale circuit simulation. Fine-grained, or procedural level problems, incur very frequent data transfers compared with the computations performed between communications [Brawer, 1989]. This is an undesirable property in parallel processing terms; from the simulation studies carried out efficiencies over fifty-percent are rare. Conversely, "coarse-grained" problems contain far less data transfers [Brawer, 1989], and any large messages can be buffered for greater efficiency, eg. computational fluid dynamics or image processing problems.

Some informative conclusions can be drawn from the small-scale system investigation, notably the effect of increased processors, computational load balancing and inter-processor communications on performance. A measure of computational load was given by the individual processor run-times required to execute the component models in each sub-circuit partition; the multi-step controller does not affect the processor load balance, as step-control takes place after the simultaneous execution of all component models. Table 6.3 details the component model run-times for the different configurations investigated, and this is reproduced graphically in Figures 6.11a to 6.11e.

Circuit partition 2(i) exhibited a poor distribution of computational effort, but required the least data transfer (one connecting transmission-line only). Re-configuration for improved processor load balancing to partition 2(ii) resulted in better performance (speed-up from 1.06 to 1.25), which more than compensates for the increased data transfer required[7]. The converse was demonstrated by the

[6] Data obtained courtesy of Transtech Devices Ltd

[7] Appendix 4 details the sub-circuit source programs for partition 2(ii)

three-processor configurations investigated. Partition 3(ii) showed an improved load balance compared with partition 3(i), but required more communications. In this case the data transfer overhead outweighed the improved arrangement, causing a reduction in performance. The change in duty-cycle input demand to the actuator controllers, for partitioned simulation 3(i), did not affect the balance of computational loads. This is demonstrated by comparison of Figures 6.12a and 6.12b, which show that the model run-times for each partition are more-or-less constant during the simulation run.

The four-processor simulation afforded no net increase in speed over the three-processor configurations, because the additional parallelism was offset completely by the very dissimilar processor loads and increased data transfers. Further partitioning to five processors, or in fact an improved four-processor arrangement, was complicated by the two feedback control loops that had to remain intact within their separate partitions. This constraint implies that a five-processor configuration is not worthwhile, because of the very significant imbalance in computational loads and the further increased communications which would inevitably result.

§6.5.2 Large-Scale System: Ring-Main Circuit

A "large-scale" hydraulic system is detailed schematically in Figure 6.13. This circuit layout is typical of the ring-main system configurations found in mobile hydraulic and marine equipment, but is used here principally to test the performance of partitioned simulation (*Burton 1994* contains the single processor *link* configuration file, which includes the physical data set). To illustrate the nature of the transient results Figure 6.14 shows the pump discharge pressure transient computed using both TLM and LSODA. The differences between the two transients are difficult to detect using a TLM pressure error of only 0.1 bar; the TLM solution is slightly more oscillatory at discontinuities, owing to the wave propagation effects included in the modelling.

Table 6.4 indicates that less computational advantage is obtained using TLM than in the small-scale system example, owing to the reduced numerical stiffness in the lumped-parameter system model of the large-scale circuit simulation. Algorithmic speed-ups from about 7 to just over 20 times were measured at different stages of the computation; TLM achieves the most significant computational gains when LSODA encounters numerical stiffness. Based on the final run-times an average TLM speed-up of approximately 12 was recorded.

§6.5.2.1 Partitioned Simulation

With much larger circuit models it is to be expected that the larger sub-circuits will increase the compute-to-communications ratio and consequently improve multi-processor efficiency.

The two, three, four and seven processor partitions are shown in Figure 6.15. The corresponding speed-ups and efficiencies are illustrated in Figure 6.16 and Figure 6.17 respectively, and detailed in Table 6.5; variations in valve and load duty-cycle inputs did not affect these results noticeably. A maximum speed-up of 2.7 was obtained using the seven-processor configuration, a marginal improvement on the four processor case; no performance data could be acquired for a T805 transputer system.

The performance of the seven-processor topology, although fastest, was severely impeded by

(i) Processor through-routing (virtual circuits)

and (ii) Less-than-ideal load balancing, as seen in Figure 6.18 (Table 6.6)

Further improvements in performance (speed-up and efficiency) might be possible if the sub-circuit partition containing the supply and metering valves could be further sub-divided. Unfortunately, this is not feasible using the present scheme, because the sub-circuit partitions would no longer map onto a master-slave processor topology directly. It is possible to restructure the algorithm slightly, but this results in much greater use of virtual circuits. In this modified scheme the "slave" sub-circuit partitions can communicate characteristic pressures directly according to the transmission lines connecting them. However, communication between the remote slaves and the master (step controller) must be handled via through-routed messages. The additional overhead and increased complexity does not make this approach an efficient, or attractive solution, although it is more flexible in terms of circuit partitioning; the domain decomposition used is not restricted to that required by a master-slave processor configuration.

The rapid loss in efficiency shown in Figure 6.17 demonstrates clearly that partitioning above four processors achieves little in performance. As expected, the efficiencies are higher for the large-scale system, owing to the coarser grained nature of the parallel computation. This is an encouraging sign, but in order to extract further potential from the parallel algorithm as it stands transputers with more links per processor are required; this may be a distinct possibility with the arrival of the six-link T9 series transputer at some future date.

§6.6 Partitioning Heuristics

The speed at which data is transferred between partitions, the relative distribution of computational loads and the proportion of compute time to communications time are all important factors in the partitioning process; inappropriate sub-circuit partitions and excessive data transfer will often negate any potential advantage in parallel computing. Particularly inefficient are through-routed, or virtual communications, which are sometimes necessary if the partitions do not map the processor configuration exactly. The latency of nearest neighbour communications is very much less and should be used wherever possible.

Computational load balancing of processors has been attempted in both of the examples studied by placing, within the topological constraints of a master-slave system, roughly equivalent numbers of component models into each sub-circuit partition. However, this approach is an oversimplification, as the difference between the processor time required by two different models can differ by up to an order of magnitude. An element of trial-and-error is therefore necessary in the partitioning process to achieve the optimum configuration.

The results of both systems studied suggest that a minimum of typically four component models per partition is required, which corresponds to the small-scale three-processor case. This gives an acceptable compute-to-communications ratio, without compromising significantly circuit division into parallel tasks. The use of virtual communications is also best avoided and as demonstrated by the (rather disappointing) large-scale system seven processor case; there is often little to be gained in using more than four processors. However, if roughly balanced partitions can be achieved then five processors are likely to give the greatest speed-up, as this corresponds to the largest master-slave configuration with nearest-neighbour data transfers.

In summary, the following guidelines should be implemented for best performance:

(i) Use nearest-neighbour processor communications wherever possible.

(ii) Choose sub-circuit partitions for balanced processor loads (without violating (i))

(iii) Do not use less than four component models per sub-circuit to maintain efficiency

(iv) Generally use a three, or four-processor configuration for near-maximum speed-up (but no more than five processors in a nearest-neighbour master-slave configuration).

It is likely that these guidelines apply more widely to a range of similar master-slave algorithms that

exhibit fine-grained parallelism, as these attributes characterise multi-step partitioned TLM.

§6.6.1 Automated Sub-Circuit Partitioning

These heuristic rules might form the basis of a parallel simulation *pre-processor* to generate, automatically, well-balanced sub-circuit partitions. This requires an estimate of all component model run-times. Subsequently, this information may be used to assess the computational effort required for a given system at every simulation step. Dividing the effort value obtained by the number of processors gives an ideal computational effort, or load, for each processor. At first the circuit may be divided arbitrarily and the estimated processor loads calculated. The difference between the ideal and measured loads then gives an indication of the computational imbalance. An optimization function, such as the total root-mean-square error (the "error" being the difference between ideal and measured processor efforts) must be minimised to obtain the best arrangement. A number of partitioning arrangements must then be evaluated by a process of iteration, without violating the topological constraints of the circuit. Penalties for data transfer may be included into the function, by modifying the processing efforts of those models responsible for inter-processor communication. Such an optimisation algorithm is sometimes referred to as "simulated annealing" and is similar in principle to those used in optimal PCB design in the electronics industry [Vecchi & Kirkpatrick, 1983].

A more effective approach might be the application of *neural networks*, which are suited to problems of non-linear optimisation. "Training" a neural network with a wide variety of circuit configurations and processor topologies may result in an efficient pre-processor. This, in effect, simulates the method the author used intuitively when configuring the examples studied in this chapter.

Currently, a less realistic option is the use of a performance *profiler* to monitor simulation speed "on-line". To implement this, a type of feedback control must be used, enabling component models to be transferred between sub-circuits automatically, as the simulation proceeds. This scheme is non-trivial and will require very substantial processing itself, which may negate any potential advantage. Moreover, the technology required for on-line profiling is very recent. In addition, the design of a non-linear optimising controller is likely to be a complex task.

§6.7 Closure

This chapter has shown that significant computational speed-ups can be obtained by restructuring hydraulic circuit simulations for use with partitioned TLM. The most substantial increase in performance was due to the TLM algorithm itself, although in the cases studied much faster simulations were possible using parallel computing. Parallel speed-ups were much less than ideal, owing to the fine-grained parallelism exhibited by partitioned TLM. Parallel efficiencies were very much worse for the small-scale system investigated compared with the large-scale system, which incorporated much larger sub-circuit partitions. This is to be expected as more computation can be achieved per time-step for similar numbers of (expensive) data transfers between partitions.

This study suggests that the partitioning of hydraulic simulations using transputers is a difficult one and that a great deal of care needs to be exercised to achieve reasonable results. An alternative parallel computing architecture might prove advantageous, such as the Intel i860-based *Quadputer*, for example. This system consists of four very fast floating point processors (approximately ten times faster than a single T805) that communicate via shared memory, as opposed to serial data-link. However, even this solution can be problematic, because the intensity of information transfer can render the shared memory system highly inefficient for systems of more than four processors. In future the six-link T9 series transputer may offer the best solution, with faster processor and communications speeds.

To conclude, the algorithmic and parallel speed-ups combined can offer considerable performance increases (something like fifty times faster) compared with the equivalent lumped-parameter computation based on a single processor using a complex, centralised ODE solver.

Simulated Time [s]	LSODA CPU Time[s]	TLM CPU Time [s]	Speed-up
0.5	55.0 (55.0)	3.1 (3.1)	17.7
1.0	2.7 (37.7)	6.6 (3.5)	10.8
1.5	100.0 (7.3)	7.5 (0.9)	8.1
2.0	857.0 (757.0)	23.0 (15.5)	48.8
2.5	864.7 (7.7)	24.2 (1.2)	5.8
3.0	872.5 (7.8)	27.2 (3.0)	2.6
3.5	877.1 (4.6)	27.5 (0.3)	15.3
4.0	822.1 (5.0)	27.8 (0.3)	16.7

[Note: Figures in brackets refer to processing times needed to compute time interval]

TABLE 6.1 TLM/LSODA ALGORITHMIC PERFORMANCE COMPARISON FOR THE CONTROLLED TWO-ACTUATOR CIRCUIT

	T800 (25MHz)			T805 (30 MHz)		
TRAMs	Run-time [s]	Speed-up	Efficiency	Run-time [s]	Speed-up	Efficiency
1	38 (41.8)	1	1	25.5	1	1
2(i)	35.8 (39.4)	1.06 (1.06)	0.53 (0.53)	-	-	-
2(ii)	30.5 (33.8)	1.25 (1.24)	0.62 (0.62)	-	-	-
3(i)	23.2 (26.2)	1.64 (1.6)	0.55 (0.53)	13.29	1.92	0.64
3(ii)	23.3 (26.2)	1.63 (1.6)	0.54 (0.53)	-	-	-
4	23.6 (26.1)	1.61 (1.6)	0.40 (0.4)	-	-	-

[Note: Roman numerals in brackets indicate different model-to-processor sub-circuit partitions. Numbers in brackets indicate results for different controller duty-cycle inputs]

TABLE 6.2 MULTI-PROCESSOR SPEED-UP AND EFFICIENCY FOR THE CONTROLLED TWO-ACTUATOR CIRCUIT

	Sub-Circuit Component Model CPU Time [s]			
TRAMs	M	S_1	S_2	S_3
1	20.2 (22.3)	-	-	-
2(i)	7.4 (8.2)	16.5 (18.2)	-	-
2(ii)	10.6 (11.7)	12.5 (13.8)	-	-
3(i)	5.7 (6.2)	9.4 (10.4)	9.3 (10.3)	-
3(ii)	6.6 (6.2)	8.7 (10.4)	8.7 (10.3)	-
4	4.0 (4.4)	8.3 (9.0)	8.2 (8.8)	2.4 (3.1)

[Note: Roman numerals in brackets indicate different partitioning. Numbers in brackets indicate results for different controller duty-cycle inputs]

TABLE 6.3 SUB-CIRCUIT PARTITION CPU TIME FOR THE CONTROLLED TWO-ACTUATOR CIRCUIT

Simulated Time [s]	LSODA CPU Time[s]	TLM CPU Time [s]	Speed-up
1.0	145 (145)	14 (14)	10.4
2.0	173 (28)	17.7 (3.7)	7.6
3.0	502 (325)	38.6 (20.9)	15.6
4.0	733 (236)	49.6 (11)	21.5
5.0	956 (228)	71.2 (21.6)	10.6
6.0	997 (41)	74.9 (3.7)	11.1
7.0	1195 (198)	95.4 (20.5)	9.7
8.0	1422 (227)	109.7 (14.3)	15.9
9.0	1468 (46)	116.5 (6.8)	6.8
10.0	1515 (47)	122.0 (5.5)	8.6

[Note: Figures in brackets refer to processing times needed to compute time interval]

TABLE 6.4 TLM/LSODA ALGORITHMIC PERFORMANCE COMPARISON FOR THE RING MAIN CIRCUIT

	T800 (25MHz)		
TRAMs	Run-time [s]	Speed-up	Efficiency
1	583.4	1.0	1.00
2	363.7	1.6	0.80
3	290.5	2.0	0.66
4	229.4	2.5	0.64
7	217.4	2.7	0.38

TABLE 6.5 MULTI-PROCESSOR SPEED-UP AND EFFICIENCY FOR THE RING MAIN CIRCUIT

	Sub-Circuit Component Model CPU Time [s]						
TRAMs	M	S_1	S_2	S_3	S_4	S_5	S_6
1	323.6	-	-	-	-	-	-
2	185.0	173.4	-	-	-	-	-
3	121.2	137.9	364.0	-	-	-	-
4	89.6	97.8	97.4	97.4	-	-	-
7	92.8	54.3	54.3	54.3	43.2	43.2	43.2

TABLE 6.6 SUB-CIRCUIT PARTITION CPU TIME FOR THE RING MAIN CIRCUIT

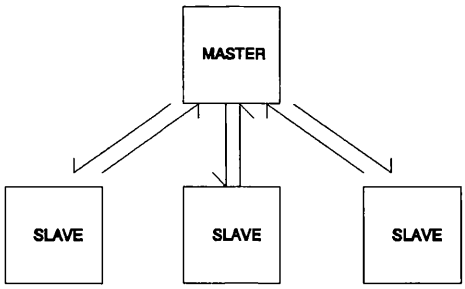
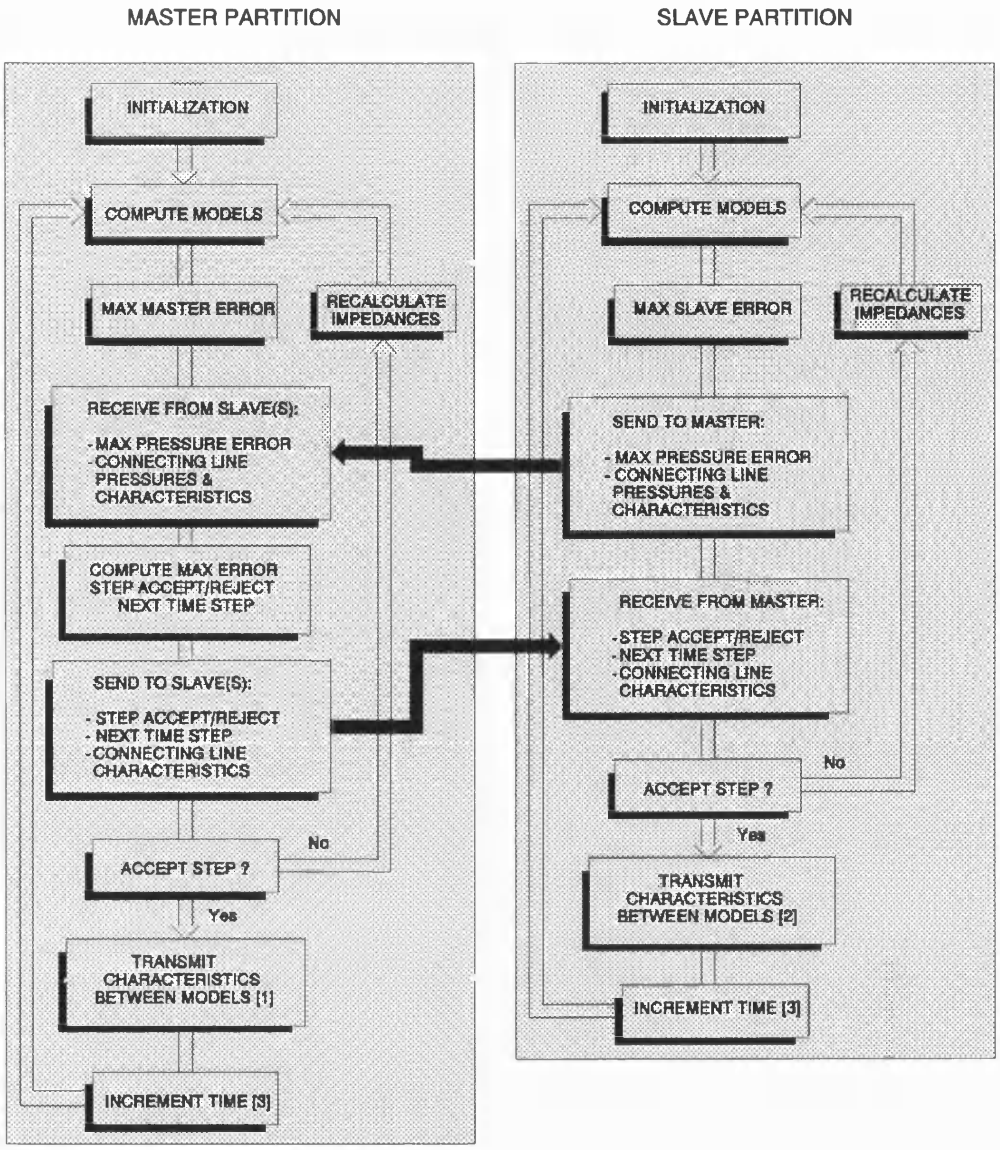


FIGURE 6.1 MASTER-SLAVE PROCESS CONFIGURATION



NOTES:

[1] INCLUDES CHARACTERISTIC PRESSURES TRANSMITTED FROM THE SLAVE PARTITION(S)

[2] INCLUDES CHARACTERISTIC PRESSURES TRANSMITTED FROM THE MASTER PARTITION

[3] FIGURE 5.2 GIVES GREATER DETAIL

LEGEND:

INTERNAL DATA TRANSFER

INTER-PROCESSOR (EXTERNAL) DATA TRANSFER

FIGURE 6.2 PARTITIONED MULTI-STEP TLM ALGORITHM

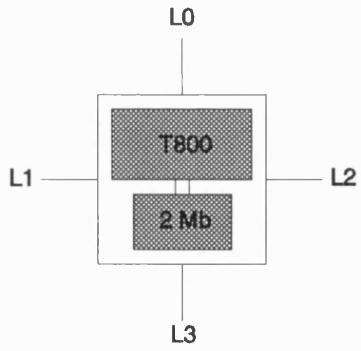


FIGURE 6.3 T800 2MB TRAM

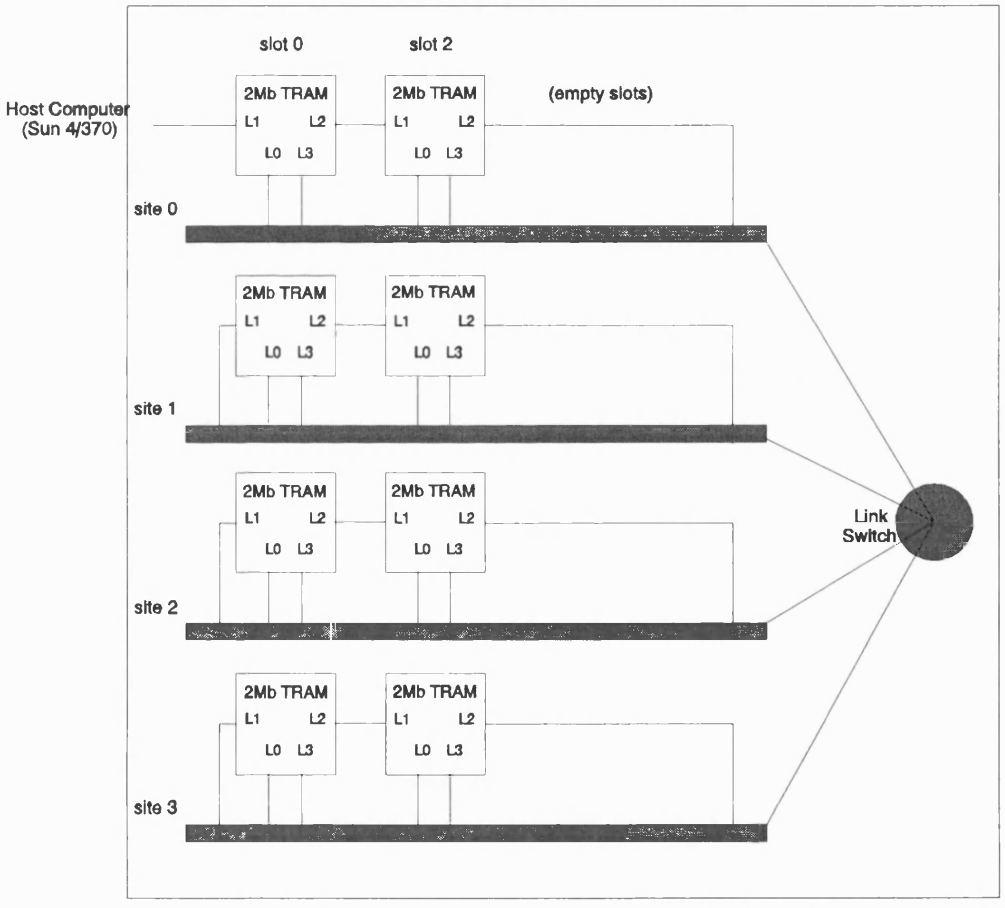


FIGURE 6.4 TRANSPUTER MOTHERBOARD (TRANSTECH MCP1000)

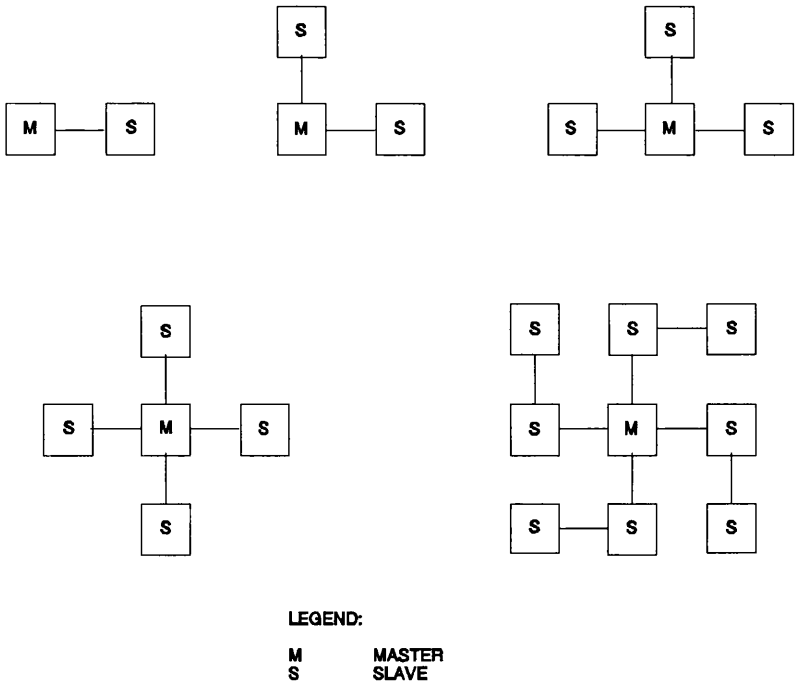


FIGURE 6.5 MASTER-SLAVE PROCESSOR TOPOLOGIES

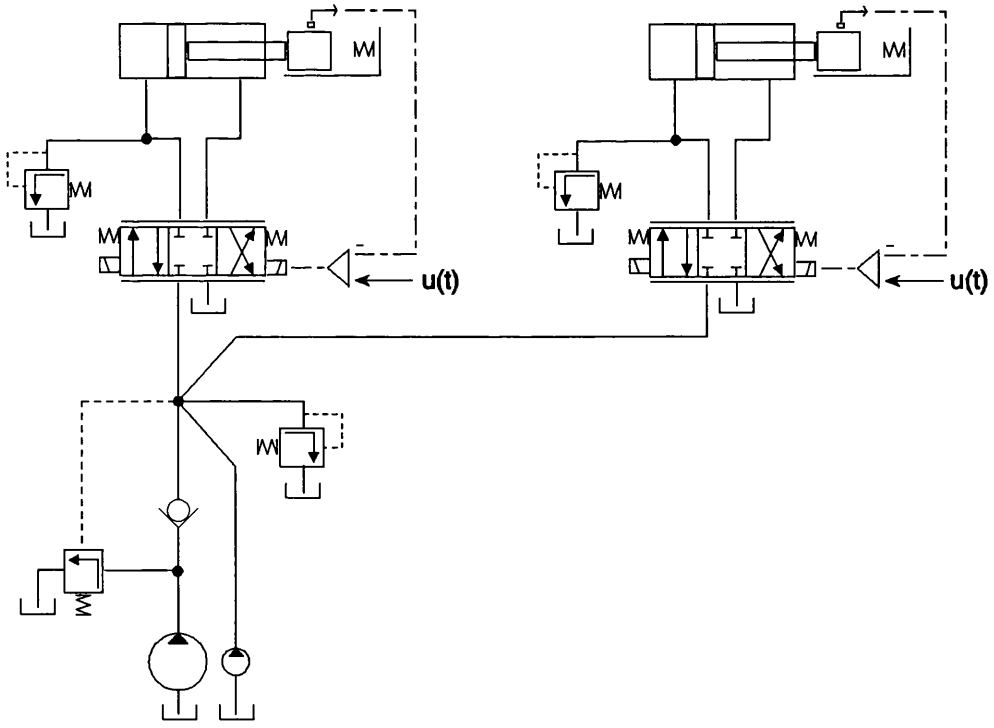


FIGURE 6.6 SMALL SCALE SYSTEM EXAMPLE: CONTROLLED TWO-ACTUATOR CIRCUIT

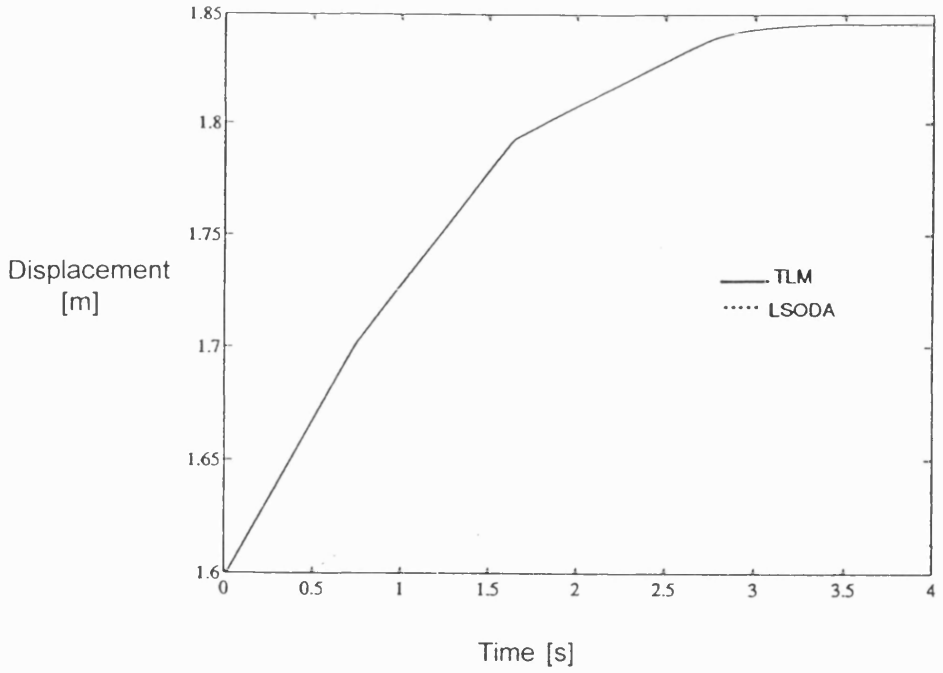


FIGURE 6.7a CONTROLLED TWO-ACTUATOR CIRCUIT: ACTUATOR DISPLACEMENT TRANSIENTS

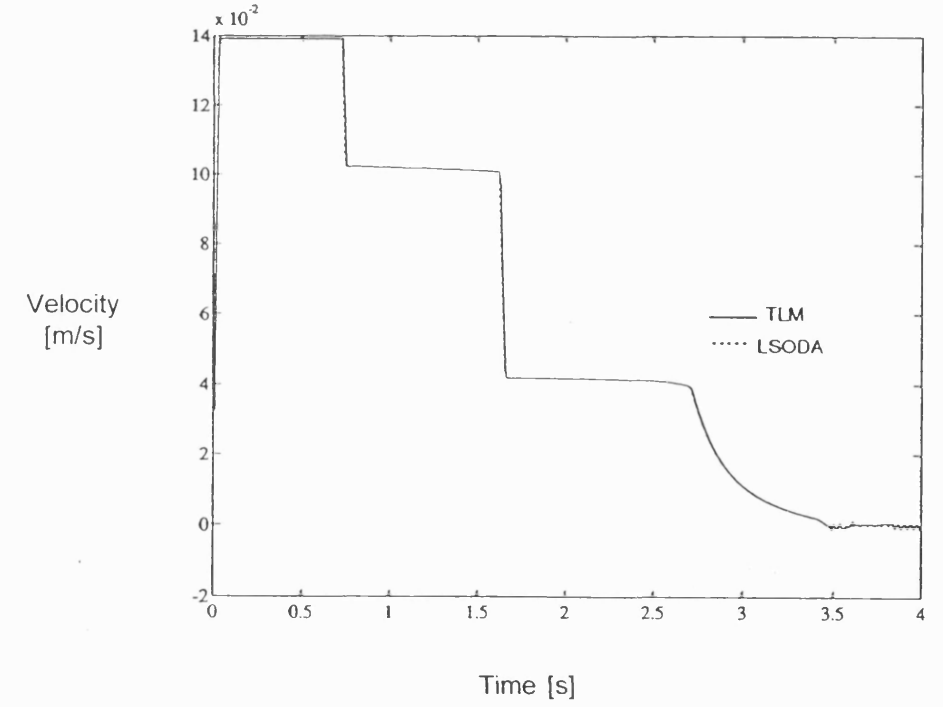


FIGURE 6.7b CONTROLLED TWO-ACTUATOR CIRCUIT: ACTUATOR VELOCITY TRANSIENTS

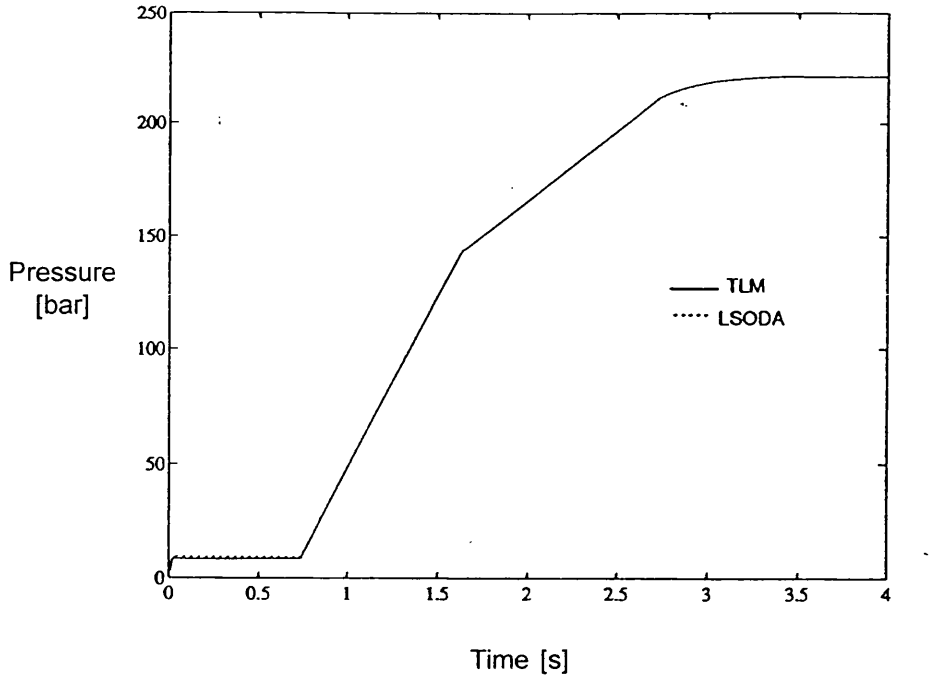


FIGURE 6.7c CONTROLLED TWO-ACTUATOR CIRCUIT: ACTUATOR INLET PRESSURE TRANSIENTS

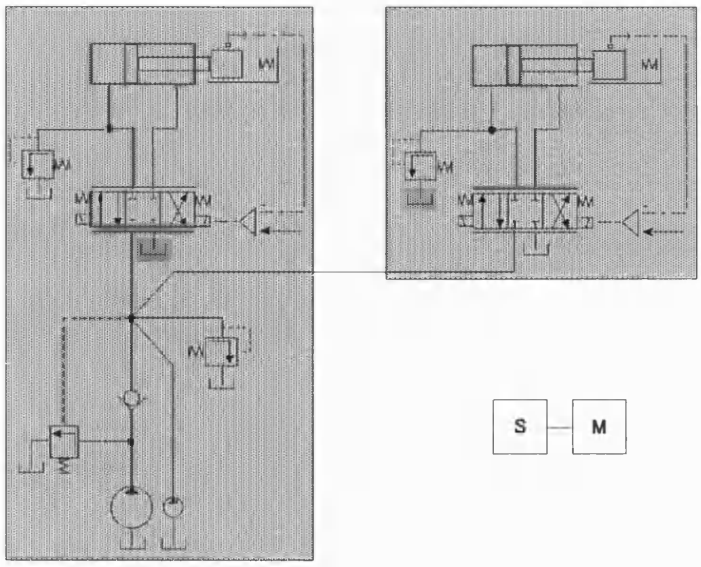


FIGURE 6.8a SMALL-SCALE SYSTEM: TWO-PROCESSOR PARTITION (i)

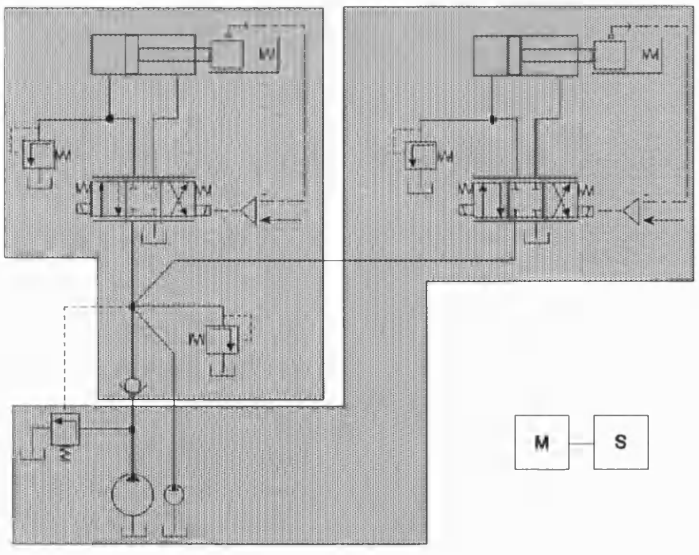


FIGURE 6.8b SMALL-SCALE SYSTEM: TWO-PROCESSOR PARTITION (ii)

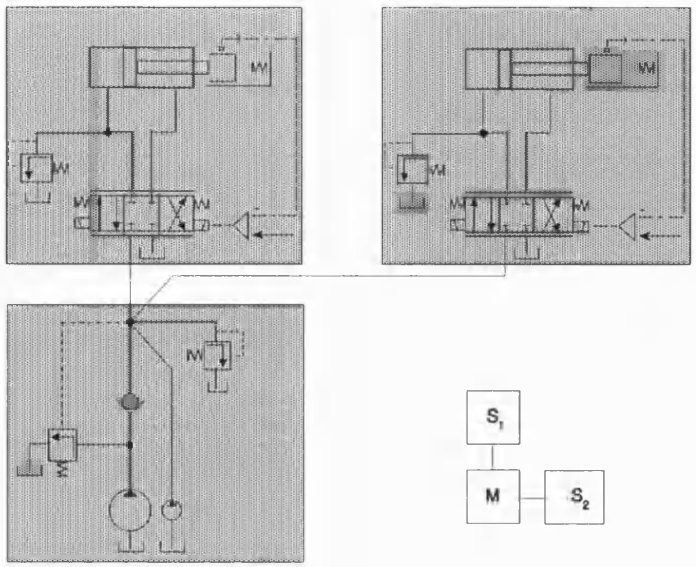


FIGURE 6.8c SMALL-SCALE SYSTEM: THREE-PROCESSOR PARTITION(i)

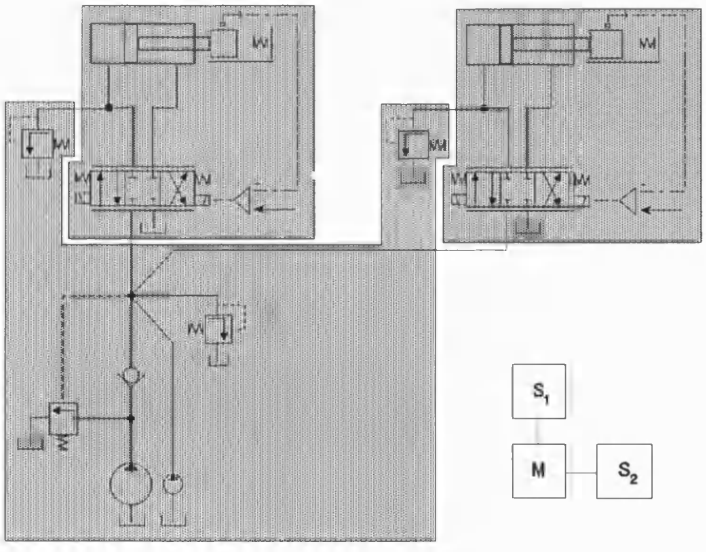


FIGURE 6.8d SMALL-SCALE SYSTEM: THREE-PROCESSOR PARTITION(ii)

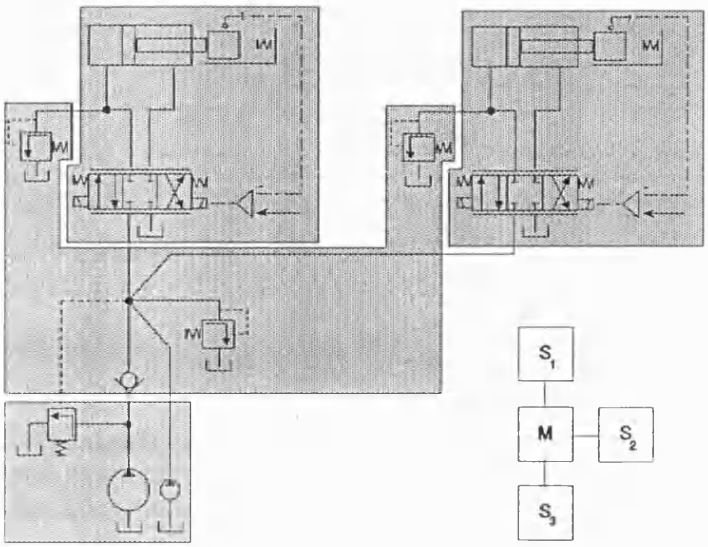


FIGURE 6.8e SMALL-SCALE SYSTEM: FOUR-PROCESSOR PARTITION

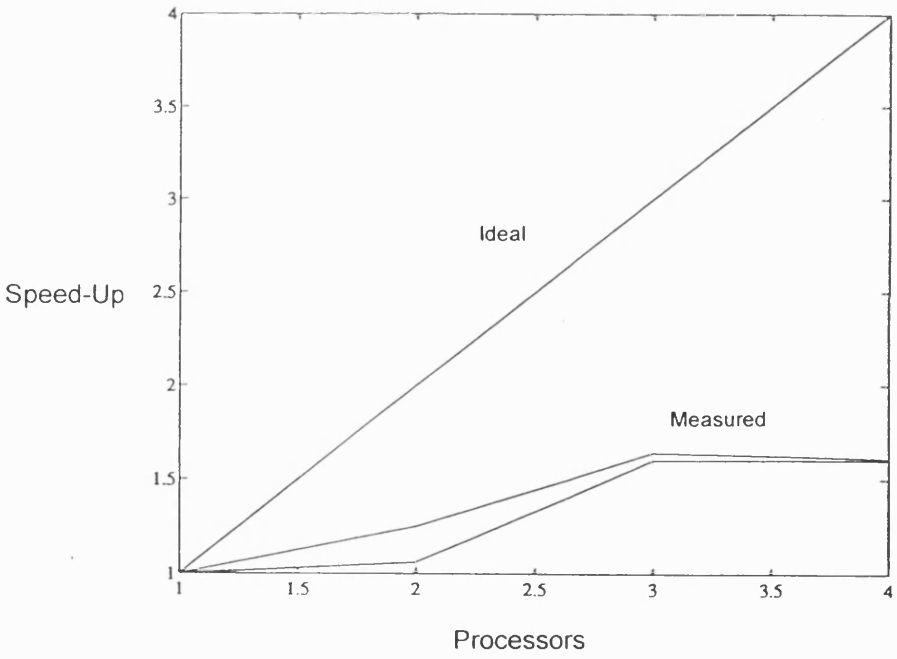


FIGURE 6.9 SMALL-SCALE CIRCUIT: SPEED-UP AGAINST PROCESSORS

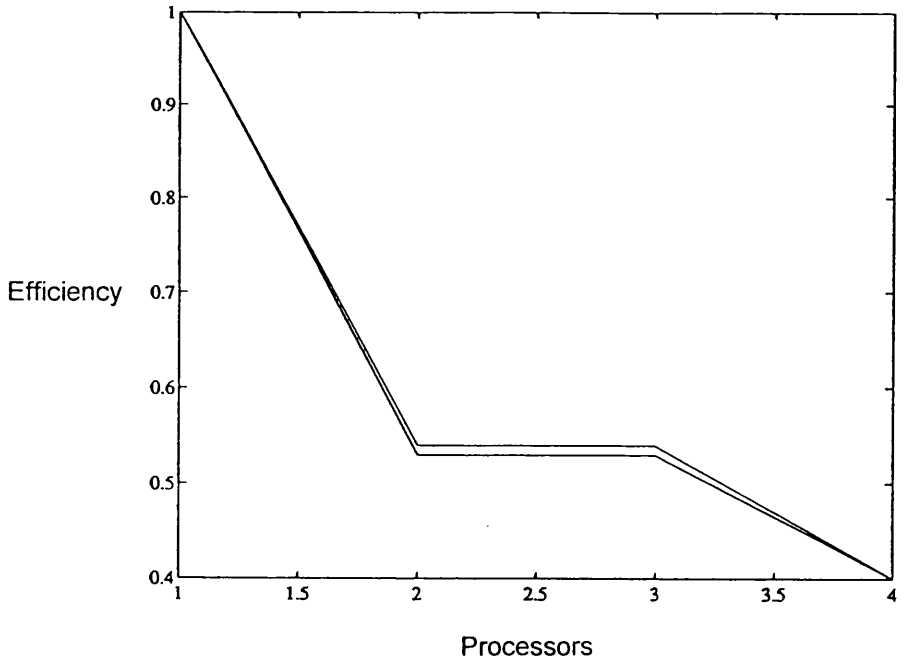


FIGURE 6.10 SMALL-SCALE CIRCUIT: PARALLEL EFFICIENCY AGAINST PROCESSORS

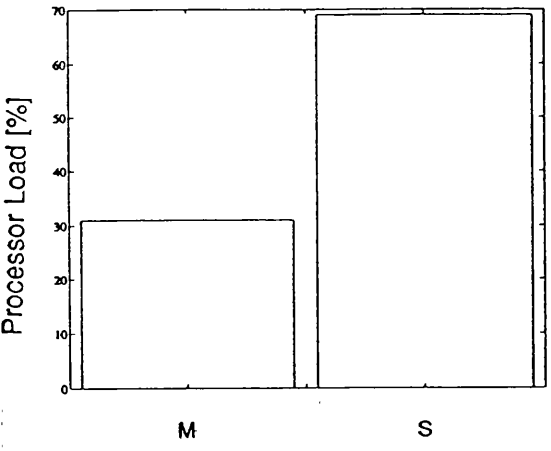


FIGURE 6.11a SMALL-SCALE CIRCUIT: COMPUTATIONAL LOAD PARTITION 2(i)

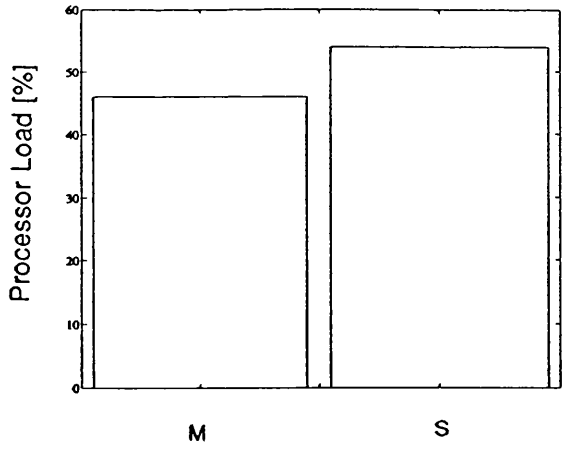


FIGURE 6.11b SMALL-SCALE CIRCUIT: COMPUTATIONAL LOAD PARTITION 2(ii)

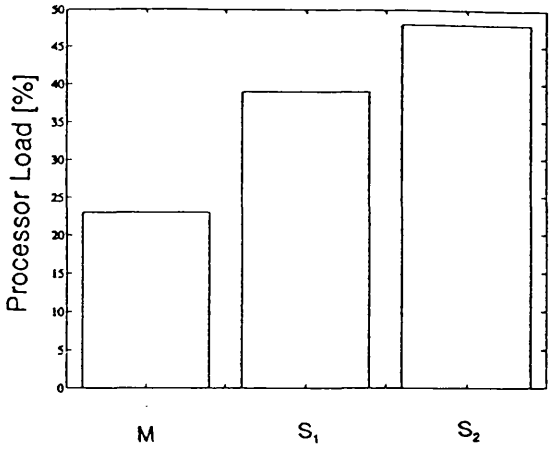


FIGURE 6.11c SMALL-SCALE CIRCUIT: COMPUTATIONAL LOAD PARTITION 3(i)

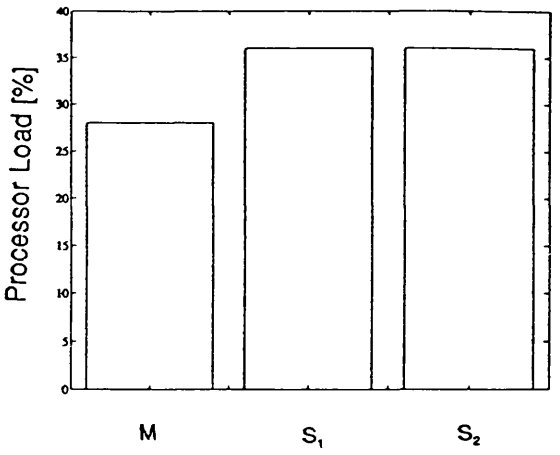


FIGURE 6.11d SMALL-SCALE CIRCUIT: COMPUTATIONAL LOAD PARTITION 3(ii)

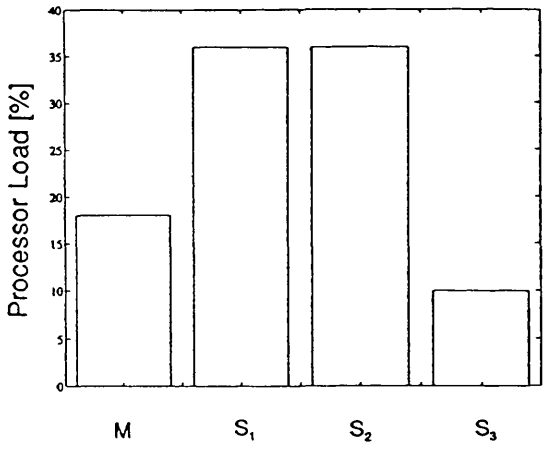


FIGURE 6.11e SMALL-SCALE CIRCUIT: COMPUTATIONAL LOAD PARTITION 4

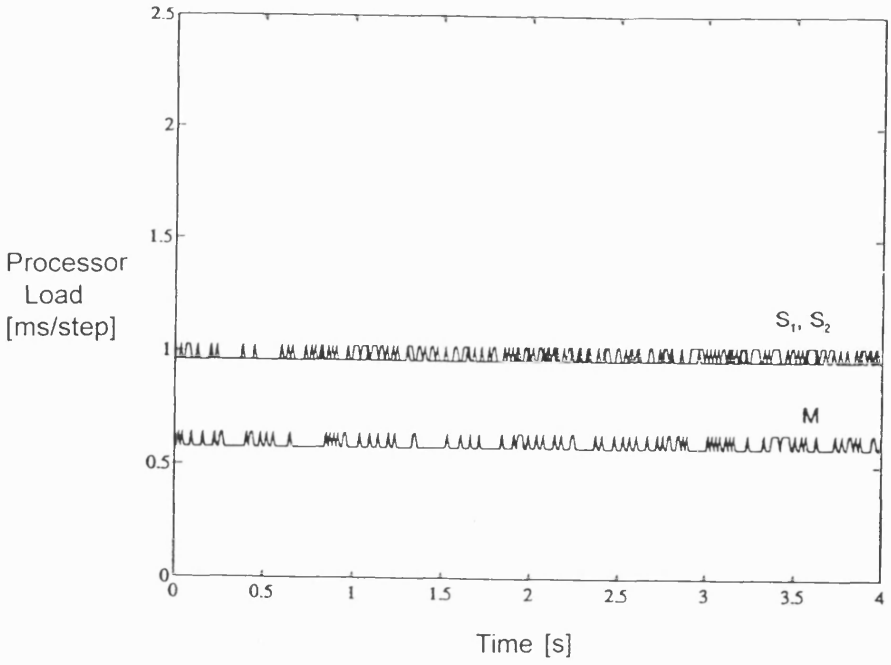


FIGURE 6.12a SMALL-SCALE CIRCUIT PARTITION 3(i): COMPUTATIONAL LOAD TRANSIENT

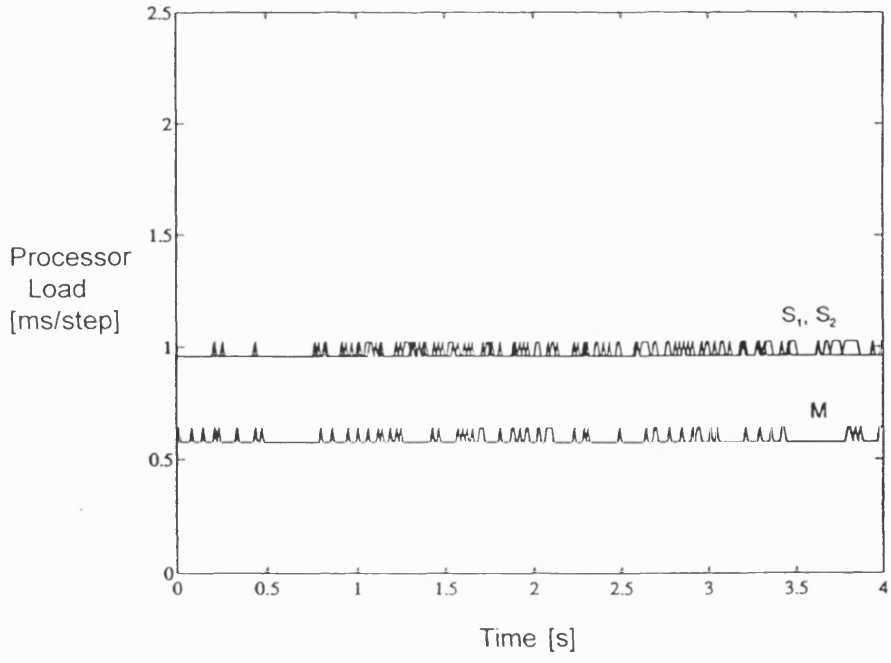


FIGURE 6.12b SMALL-SCALE CIRCUIT PARTITION 3(ii): COMPUTATIONAL LOAD TRANSIENT

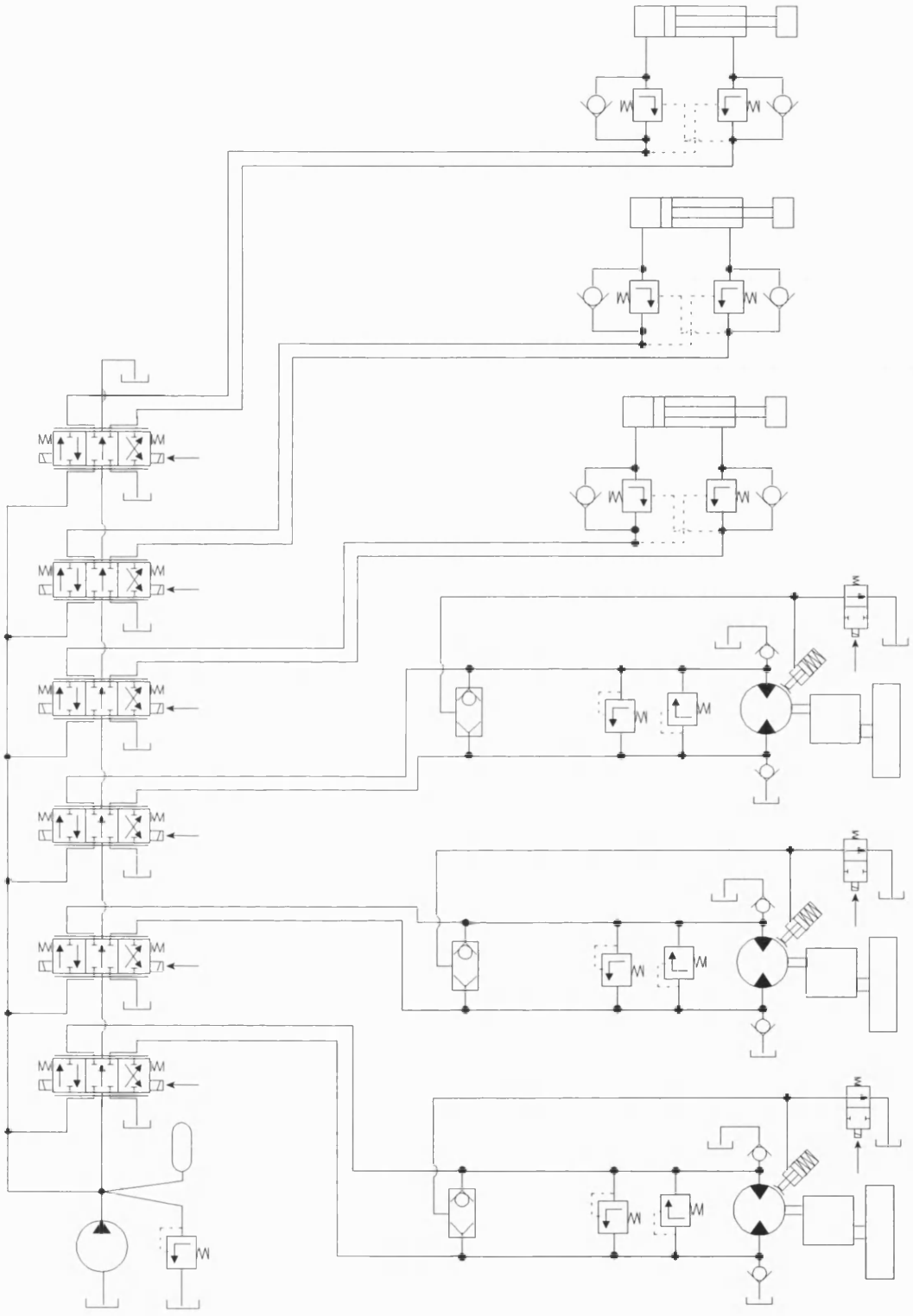


FIGURE 6.13 LARGE-SCALE SYSTEM EXAMPLE: RING-MAIN CIRCUIT

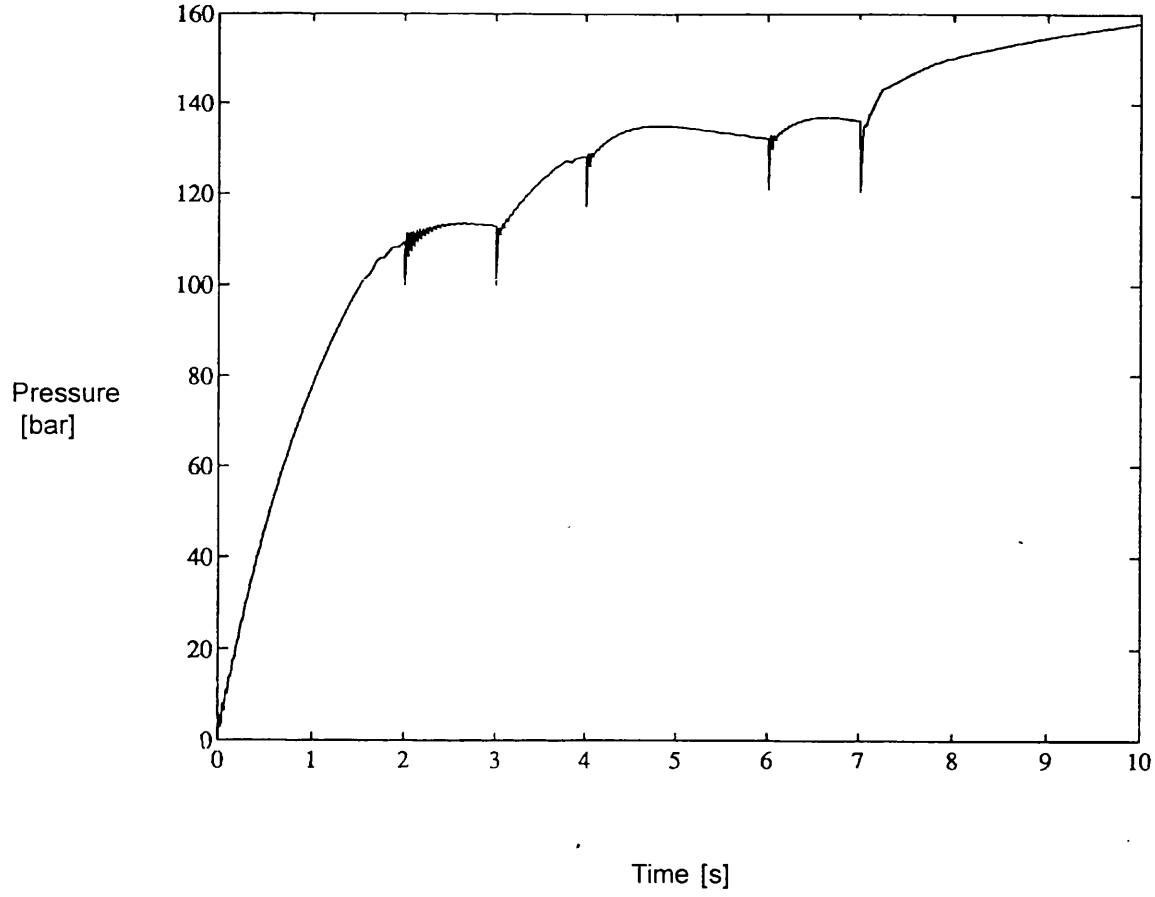


FIGURE 6.14 RING-MAIN SYSTEM: PUMP DISCHARGE PRESSURE TRANSIENT

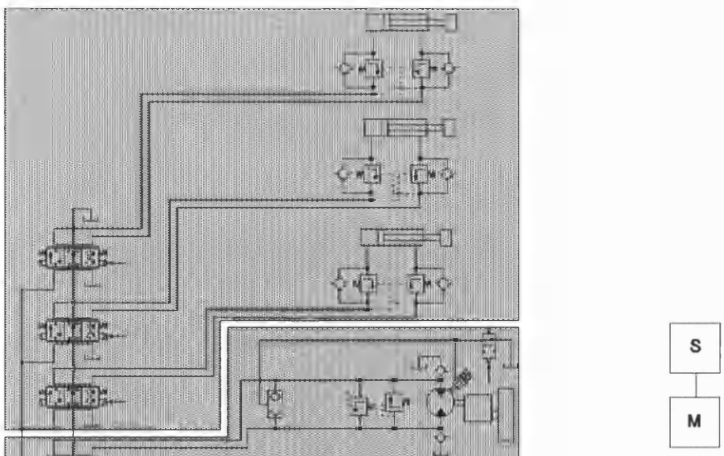


FIGURE 6.15a LARGE-SCALE SYSTEM: TWO-PROCESSOR PARTITION

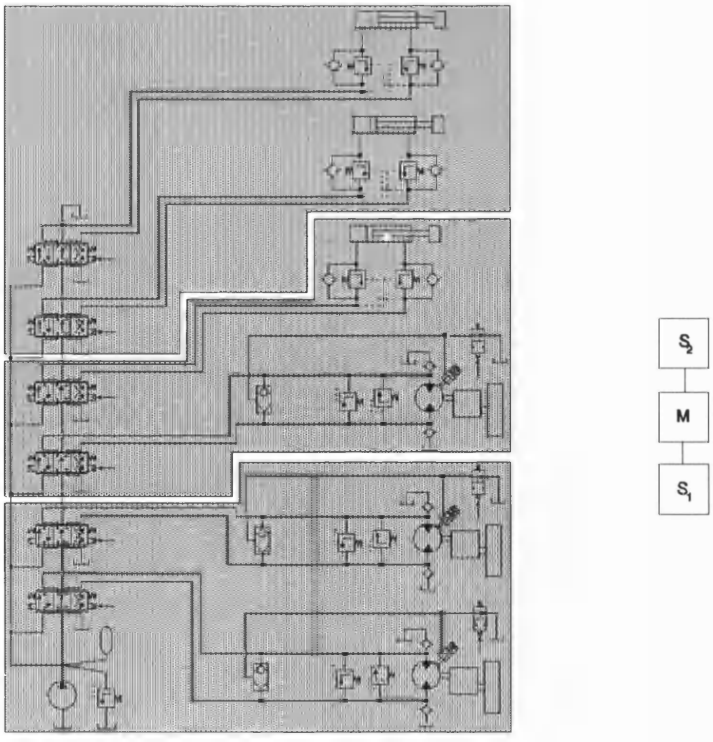


FIGURE 6.15b LARGE-SCALE SYSTEM: THREE-PROCESSOR PARTITION

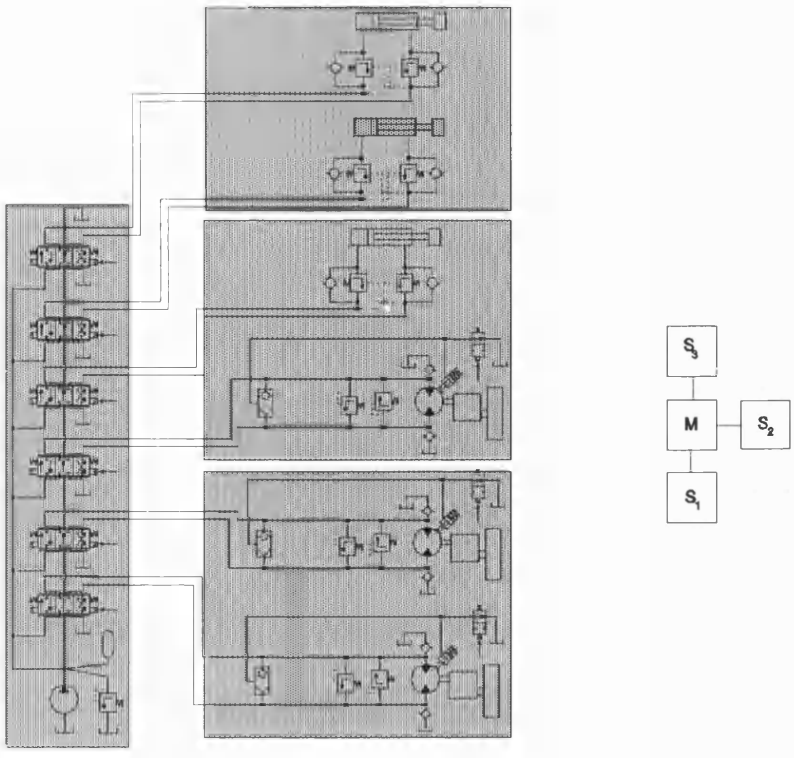


FIGURE 6.15c LARGE-SCALE SYSTEM: FOUR-PROCESSOR PARTITION

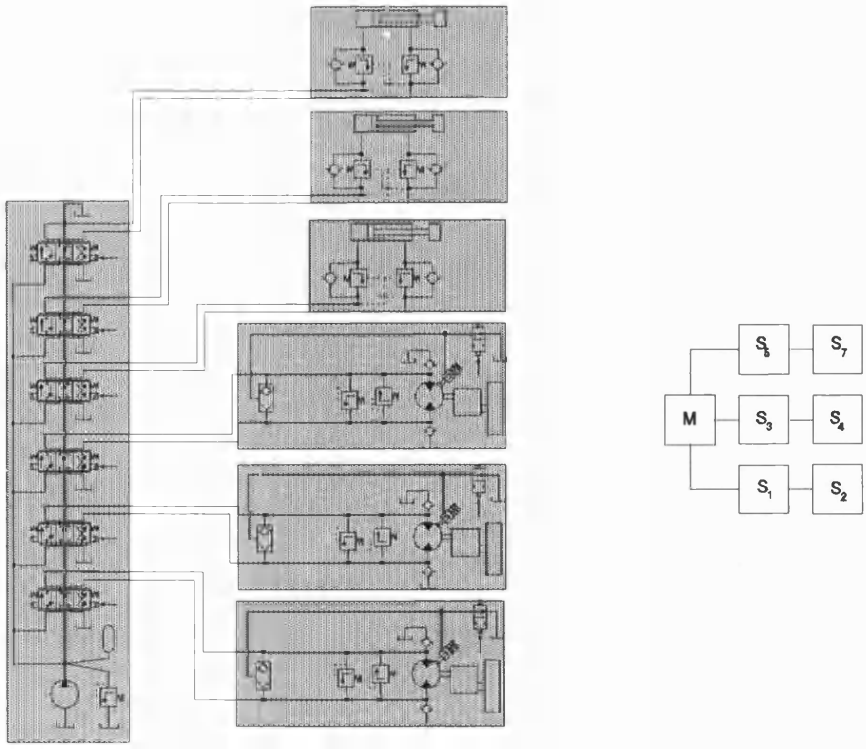


FIGURE 6.15d LARGE-SCALE SYSTEM: SEVEN-PROCESSOR PARTITION

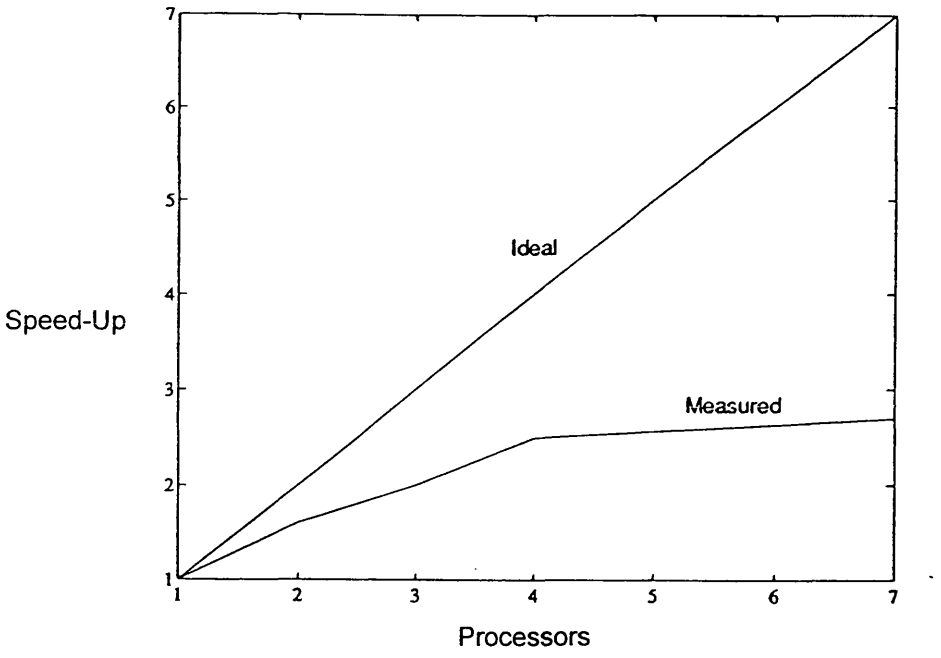


FIGURE 6.16 LARGE-SCALE CIRCUIT: SPEED-UP AGAINST PROCESSORS

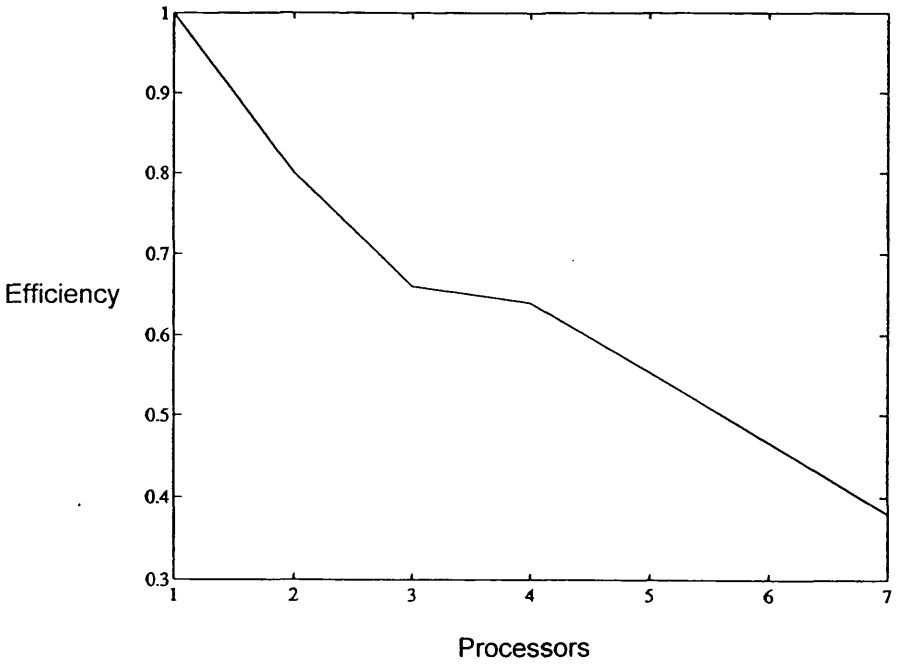


FIGURE 6.17 LARGE-SCALE CIRCUIT: PARALLEL EFFICIENCY AGAINST PROCESSORS

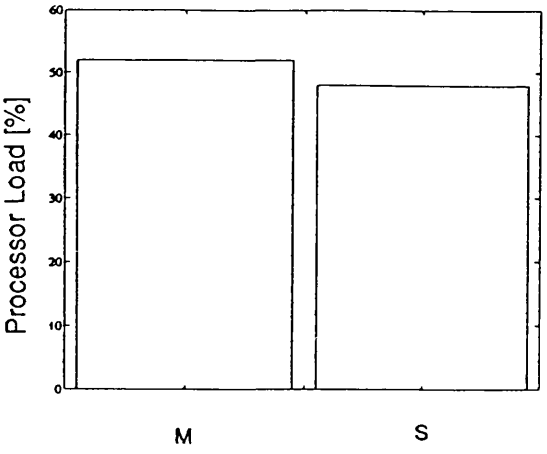


FIGURE 6.18A LARGE-SCALE CIRCUIT: COMPUTATIONAL LOAD PARTITION 2

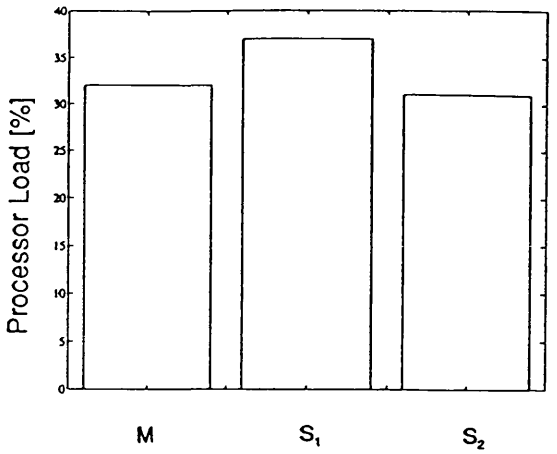


FIGURE 6.18b LARGE-SCALE CIRCUIT: COMPUTATIONAL LOAD PARTITION 3

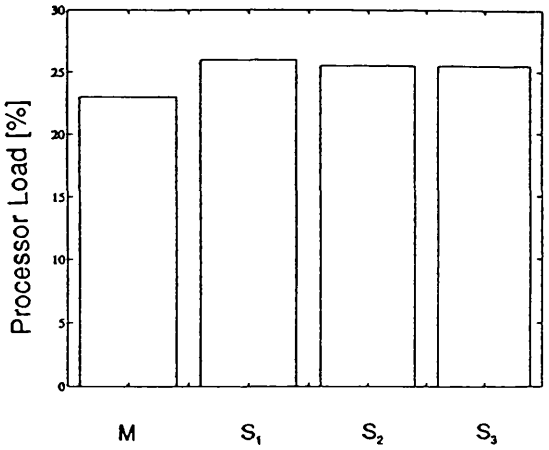


FIGURE 6.18c LARGE-SCALE CIRCUIT: COMPUTATIONAL LOAD PARTITION 4

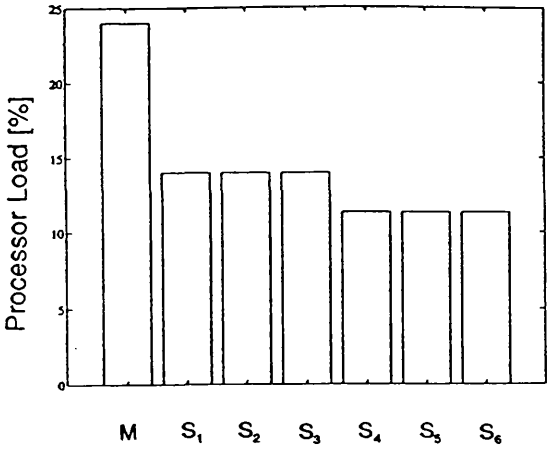


FIGURE 6.18d LARGE-SCALE CIRCUIT: COMPUTATIONAL LOAD PARTITION 7

CHAPTER 7

CONCLUSIONS

§7.1 Summary of Conclusions

This thesis has demonstrated the application of distributed-parameter transmission-line modelling (TLM) techniques to complex hydraulic system simulation. A finite time delay is introduced by the fluid transmission-lines to model the physical wave propagation delay, which allows the decoupling of individual circuit elements (component models) in a system simulation. It is this property of the hydraulic transmission-line that enables a system model to be sub-divided and hence computed in parallel. Multi-processor operation of the lumped-parameter simulation was not considered worthwhile, owing to the very considerable processing time allocated to central integration, and the highly sequential series of numerical operations required.

Very significant improvements in execution time have been achieved using TLM compared with the conventional approach to lumped-parameter modelling using piece-wise continuous ODEs. A stiff numerical computation that required some 4860 seconds of processor time was simulated in only 189 seconds using an equivalent transmission-line circuit model. The differences between the results for the different numerical computations were typically within one percent. Multi-step TLM in conjunction with the optimum parallelisation of a TLM circuit model into sub-systems can effect combined (algorithmic and parallel processing) speed-ups approaching fifty times. Lumped-parameter simulations that are "stiff" when solved using conventional means of numerical integration, gave the most substantial performance improvements when restructured for transmission-line modelling.

The even distribution of processor loads was found to be an important criteria for partitioned simulation. Incorrect load balancing reduced the maximum speed up achievable (equal to the number of processors employed), irrespective of the transmission delay associated with sending messages

between the different circuit partitions at the end of each time step. "Large-scale" circuit models were discovered to be more amenable to efficient sub-division. This is because (i) there are a greater number of component models in each partition for a given number of processors and (ii) there is greater flexibility in the choice of partitioning arrangement. These considerations can increase the compute to communicate ratio and hence parallel efficiency.

It is the high frequency of inter-processor communications, the "fine-grained" parallelism, that limits the efficiency to somewhere between fifty and seventy-five percent for the fastest partitioned simulations. Moreover, the efficiency of parallel computing can reduce so rapidly, that a four-processor simulation, giving a speed-up between two and three times, is about the maximum attainable for the types of problem usually encountered in fluid power simulation.

§7.2 Present Status

A reconfigurable, model-based simulation program has been developed using transmission-line modelling (TLM) to analyze the transient response of arbitrary hydraulic system configurations. Individual component model equations are solved simultaneously with the transmission-line equations at each fluid connection, such that every model is self-contained and numerically isolated by a finite transmission delay. Differential equations for components may be converted into finite difference form using bi-linear transformation (equivalent to trapezoidal integration) within the component model. This approach represents a truly distributed solution technique, that is theoretically well suited to parallel computation.

Multi-processor simulation can be achieved by dividing the circuit model into a number of connected sub-circuit models. Concurrent operation is possible, as the inputs to each sub-circuit partition are delayed outputs from connected partitions. This is the only criteria necessary for a fixed step TLM solution. However, the use of an efficient multi-step TLM solver imposes certain limitations on the way in which a circuit model can be sub-divided. To achieve sub-system synchronisation and accuracy control with a solver that uses variable time-stepping, a "master-slave" processor topology must be employed. The master process, in addition to computing a sub-circuit, collates accuracy information from all partitions, in order to set the simulation time step and transfer this information to slave processes.

Instantaneous, uni-directional signals can be propagated between component models by appropriate

model call sequencing. This is an important development in the simulation of systems that employ feedback control, for example. There are further implications for circuit partitioning, because the "chain" of component models connected by (instantaneous) signal links cannot be broken, forcing the affected models into a single sub-circuit. The difference in wave speed between electrical (speed of light) and hydraulic transmission lines (speed of sound in fluid) means that the finite delay between electrical components is incredibly small compared with the hydraulic components. The use of instantaneous uni-directional links enables realistic propagation delays between the hydraulic parts of a system without introducing unacceptable delays into the electrical parts.

This parallel processing scheme often results in many parallel configurations, both in terms of the number of processors used and the exact nature of the circuit sub-division. Different arrangements can lead to widely differing performance. Attention must be given to the distribution of processor loads, in addition to the frequency of (inefficient) communications between them.

Computer simulations using the multi-step TLM solver are initiated by a process of program generation. This involves automatically creating executable files using a purpose-designed program generator. In its present form the program generator requires input from an ASCII data file, which contains details of the standard component models used, their connectivity and associated parametric data. Each component model must be selected from those contained within an expandable model library, which now consists of some fifty-five items (see "model attributes" file detailed in *Burton 1994*). After checking to ensure valid circuit connectivity, code generation of the control program, compilation and linking with the relevant models all takes place automatically to produce an executable file.

Single processor program generation is straightforward to accomplish using this approach. Multi-processor operation, by its very nature, is considerably more complex to achieve. Each sub-circuit must first be generated independently and subsequently linked using a pre-defined configuration script. This script file must contain details of the exact processor topology already established, in order to map the respective circuit partitions onto the correct processors.

§7.3 Evaluation

Significant algorithmic speed-ups have been demonstrated using a multi-step implementation of the TLM solver, compared with an equivalent lumped-parameter system, modelled using ODE's.

Typically, for "stiff" lumped-parameter systems the simulation performance increase may be as much as twenty, or thirty times for comparable accuracy. Although highly system dependent, this represents a substantial improvement even without the use of parallel computing.

More modest multi-processor TLM speed-ups have been recorded, compared with the equivalent single-processor TLM computation. The balance of processor loads during parallel operation and the frequency of communications between the partitions has a very detrimental effect on performance. There may be some scope for reducing the communications between partitions to alleviate this problem, provided simulation accuracy is not affected too severely. For example, there may be certain easily identifiable sub-systems that are only weakly coupled by relatively large capacitive pipe volumes, and less frequent data transfer between such partitions will improve the performance of the parallel computation.

In general, the "fine-grained" parallelism of TLM problems means that efficiencies over fifty percent are difficult to achieve using the T800 transputer system, owing to frequent communications. Large-scale system simulations, incorporating many more component models, are more disposed to efficient operation and consequently greater speed-ups. This is because of the much larger sub-circuit partitions that can be established for a given number of processors, resulting in a more efficient "coarse-grained" simulation. Clearly, there is a finite limit to the number of processors that can be applied to any given problem without data transfer tending to dominate the computational effort.

In the case of the small-scale system investigated (23 component models) a maximum speed-up of 2 was obtained using a four processor configuration. A maximum speed-up of 2.7 was demonstrated by the large-scale system studied (85 component models) using seven processors. This latter configuration required the use of highly inefficient "through-routed" message passing, ie. messages sent between two processors via an intermediate processor, owing to the limitation of the T8 series transputer to four serial data links. The four processor implementation of this circuit gave only a slightly reduced speed-up of 2.5.

§7.4 Directions for Future Research

Throughout the progress of this research a number of areas for further investigation and development were found. In the short-term this should include an increase in the diversity of component and transmission-line models (although new models are more often developed as the need arises). An area

of more substantial work is a fully automated program generator for parallel computing.

At the inception of this research project the transputer was ostensibly the only parallel device on the market. Technological development over the last few years has lead to several alternatives that may be more disposed to efficient operation, given the often fine-grained nature of TLM simulations. An example is the Inmos i860 Quadputer, which as the name implies contains four extremely powerful floating point processors, linked via shared memory, as opposed to serial data link.

For a given platform, the very subjective area of automated sub-circuit partitioning should be tackled. Initially, this may take the form of a pre-processor, employing non-linear optimisation techniques to disseminate the circuit model. Allowing for development of the appropriate technology, the most efficient operation may be achieved "on-line" via direct performance monitoring of individual processors and automated reconfiguration, if necessary.

On-line condition monitoring and sophisticated control techniques using TLM reference simulations for real plant are other important areas worthy of investigation. Fixed step TLM may be the only way to guarantee predictable performance in terms of execution time, although not always in terms of accuracy. The actual time taken to compute a given time interval can vary considerably for multi-step TLM, from faster than real-time to slower than real-time, depending upon the nature of the external disturbance(s) and the component models used.

For on-line condition monitoring, non-linear parameter identification techniques may provide the necessary means to determine system integrity and to warn of incipient faults. In terms of advanced control, assuming that a TLM reference model can be made sufficiently fast and accurate, there still exists the problem of exactly how to use the reference states, derived from a non-linear observer, in order to control the plant in a prescribed way.

REFERENCES

Auslander D M, 1968, "Distributed System Simulation with Bilateral Delay-Line Models", *Trans. ASME J. Basic Eng.*, Vol 90, pp195-200.

Boucher R F & Kitsios E, 1986, "Simulation of Fluid Network Dynamics by Transmission-line Modelling", *Proc. IMechE*, Vol 200, N° C1, pp21-29.

Blackburn J F, Reethof G & Sheerer J L, 1960, "*Fluid Power Control*", MIT Press, p138.

Brawer S, 1989, "*Introduction to Parallel Programming*", Academic Press Inc., pp3-4

Brown F T, 1962, "The Transient Response of Fluid Lines", *Trans. ASME J. Basic Eng.*, Vol 84, pp547-553.

Burton J D, 1990, "Distributed Simulation of Fluid Power Systems: Preliminary Research", *University of Bath*, School of Mechanical Engineering Report N°1083

Burton J D, 1993, "Distributed Simulation of Hydraulic Systems Using Transmission-Line Modelling", *University of Bath*, School of Mechanical Engineering Report N°010/93

Burton J D, 1994, "TLM Code for Parallel Simulation of Hydraulic Systems", *University of Bath*, School of Mechanical Engineering Report No. 031/1994.

Burton J D, Edge K A & Burrows C R, 1992, "Modelling Requirements for the Parallel Simulation of Hydraulic Systems", *ASME Winter Annual Meeting*, Anaheim, California, USA. 92-WA/FPST-11

Burton J D, Edge K A & Burrows C R, 1993A, "Analysis of an Electro-Hydraulic Position Control Servo-System Using Transmission-line Modelling", *2nd JHPS International Symposium on Fluid Power*, Tokyo, JAPAN.

Burton J D, Edge K A & Burrows C R, 1993B, "Partitioned Simulation of Hydraulic Systems Using Transmission-line Modelling", *ASME Winter Annual Meeting*, New Orleans, LA, USA. 93-WA/FPST-4

Burton J D, Edge K A & Burrows C R, 1993C, "Computational Load Balancing of Parallel Hydraulic Circuit Simulations Employing Variable Timestep Transmission-line Modelling", *3rd Scandinavian Conference on Fluid Power*, Linköping, SWEDEN.

Carslaw H S & Jaeger J C, 1963, "*Operational Methods in Applied Mathematics*", Dover Publications inc., New York, pp184-209

Craig J, 1988, "*Adaptive Control of Mechanical Manipulators*", Addison-Wesley, p47.

Davison P, Longmore D K & Burrows C R, 1993, "Generation of Complex Mechanical Loads in Hydraulic System Applications", *3rd Scandinavian International Conference on Fluid Power*, Vol 2, pp13-27.

D'azzo J & Houpis C H, 1988, "*Linear Control System Analysis and Design*", 3rd Ed., Mc Graw-Hill International.

Donne M S, Tilley D G & Richards C W, 1993, "Optimisation Techniques in Fluid Power Systems Design", submitted to *Research in Engineering Design*, Springer-Verlag.

D'souza A F and Oldenburger R, 1964, "Dynamic Response of Fluid Lines", *Trans. ASME J. of Basic Eng.*, Vol 86, pp589.

Elleman A U, Lindberg I, Vilenuis M J, 1993, "Simulation in the Design of Hydraulic Driven Machines: New Approach and Aspects", *3rd Scandinavian International Conference on Fluid Power*, Vol 2, pp29-41.

Fox J A, 1977, "*Hydraulic Analysis of Unsteady Flow in Pipe Networks*", MacMillan Press Ltd, Hong Kong.

Goodson R E & Leonard R G, 1972, "A survey of Modelling Techniques for Fluid Line Transients", *Trans. ASME J. Basic Eng.*, Vol 94, pp474-482.

Hairer E & Wanner G, 1991, "*Solving Ordinary Differential Equations II, Stiff and Differential Algebraic Problems*". Springer-Verlag, Berlin Heidelberg.

Handroos H M, 1991, "A Method for Postulating Flexible Models for Individual Components for Fluid Power Circuit Performance Simulation", *Proc. FLUCOME '91*, ASME, San Francisco, USA, pp519-525.

Harris R M, 1990, "The Modelling and Simulation of Temperature Effects in Hydraulic Systems", *PhD Thesis*, University of Bath, UK.

Hockney R W & Jesshope C R, 1981, "*Parallel Computers: Architecture, Programming & Algorithms*", Adam Hilger Ltd, Bristol, pp4-5,pp14-16

Hunt T M, 1986, "A Review of Condition Monitoring Techniques Applied to Fluid Power Systems", *7th International Fluid Power Symposium*, University of Bath, UK.

Innis G & Rexstad E, 1983, "Simulation Model Simplification Techniques", *Simulation*, Society for

Computer Simulation (SCS), Vol 41, July 1983.

Isermann R, 1991, "Fault Diagnosis of Machines via Parameter Estimation and Knowledge Processing", *IFAC/IMACS Symposium on Fault Detection, Supervision & Safety for Technical Processes, SafeProcess '91*, Baden-Baden, Vol 1, pp121-133.

Jansson A, Krus P & Palmberg J-O, 1992, "Variable Time Step Size Applied to Simulation of Fluid Power Systems Using Transmission Line Elements", *Proc. 5th Bath International Fluid Power Workshop*, Research Studies Press, UK.

Johns P B, 1977, "A Simple Explicit and Unconditionally Stable Numerical Routine for the Solution of the Diffusion Equation", *International Journal of Numerical Methods in Engineering*, Vol 11, pp1307-1328

Johns P B & O'Brien M, 1980, "Use of the Transmission Line Modelling (TLM) method to Solve Non-linear Lumped Networks", *The Radio and Electronic Engineer*, Vol 50, No 1/2, pp59-70.

Karam J T, 1972, "A Simple Time Domain Model For Fluid Transmission Line Systems", *Dphil Thesis, Purdue University*, Lafayettte, USA.

Karam J T, 1974, "Two Port Models of Column Separation in Line Systems Containing An Aerated Liquid", *Tech. Brief Trans. ASME J. Fluids Eng.*, Vol 96(4), pp299-301.

Karam J T & Leonard R G, 1973, "A Simple Yet Theoretically Based Time Domain Model For Fluid Transmission Line Systems", *Trans. ASME J. Basic Eng.*, Vol 95, pp498-504.

Karam J T & Tindall R, 1975, "A Refinement of A Simplified Model For Transients in Fluid Lines", *Trans. ASME J. Fluids Eng.*, Vol 97, pp376-377.

Kitsios E & Boucher R F, 1985, "Stability Analysis of Mixed Lumped-Distributed Parameter (Delay) Systems", *Journal of the Franklin Institute*, Sept/Oct 1985, pp133-150.

Kojima E & Shinada M, 1984, "Fluid Transient Phenomena Accompanied With Column Separation in Fluid Power Pipelines", *Bulletin, JSME* Vol 27 N° 233.

Krus P, Jansson A, Palmberg J-O & Weddfelt K, 1990, "Distributed Simulation of Hydromechanical Systems", *Proc. 3rd Bath International Fluid Power Workshop*, Research Studies Press, UK.

Krus P, Weddfelt K & Palmberg J-O, 1991A, "Optimisation for Component Selection in Hydraulic Systems", *Proc. 4th Bath International Fluid Power Workshop*, Research studies Press, UK, pp217-

231.

Krus P, Weddfelt K & Palmberg J-O, 1991B, "Fast Pipeline Models for the Simulation of Hydraulic systems", *Proc. ASME WAM*, Atlanta.

Krus P, Jansson A & Palmberg J-O, 1991C, "Real Time Simulation of Hydraulic Control Systems with Complex Mechanical Loads", *Linköping Studies in Science and Technology*, Thesis N°295, Department of Mechanical Engineering.

Lambert J D, 1973, "*Computational Methods in Ordinary Differential Equations*", Wiley, New York.

McCandlish D & Dorey R E, 1983, "The Mathematical Modelling of Hydrostatic Pumps and Motors", *Proc. IMechE*, Vol 198B, N°10, pp165-174.

Palmberg J O, 1991, "Fluid Power Engineering: A Field for Computer Applications", *Proc. 2nd International Conference on Fluid Power*, Tampere, Finland, pp415-431.

Partridge G J, Christopoulis C & Johns P B, 1987, "Transmission Line Modelling of Shaft System Dynamics", *Proc. IMechE* Vol. 201 C4, pp271-278.

Paygude D G, Vasudeva Rao B & Joshi S G, 1985, "Fluid Transients Following Valve Closure by FEM [Finite Element Method]", *Proc. International Conference on Finite Elements in Computational Mechanics*, Bombay, India

Petzold L, 1983, "Automatic Selection of Methods for Solving Stiff and Non-stiff Systems of ODEs", SIAM J. Scientific Statistics & Computing, Vol 4, pp136-148.

Pulko S H, Mallik A, Allen R & Johns P B, 1990, "Automatic Time-stepping in TLM Routines for the Modelling of Thermal Diffusion Processes", *International Journal of Numerical Modelling: Electronic Networks, Devices & Fields*, Vol 3, pp127-136

Pulko S H & Olashore A O, 1989, "Modelling of Heat Transfer From Fluid Sources Using The TLM Technique", *International J. Modelling & Simulation*, Vol 9, N°4, pp96-100

Richards C W, Tilley D G, Tomlinson S P & Burrows C R, 1990, "Type Insensitive Integration Codes for the Simulation of Fluid Power Systems", *Proc. ASME WAM*, Texas, USA.

Sato S, Kume S & Kobayashi K, 1991, "Non-linear Observer of Electro-Hydraulic Servomotor", *Proc. 3rd Bath International Fluid Power Workshop*, Research Studies Press, UK.

Sidell R S & Wormley D N, 1977, "An Efficient Simulation Method for Distributed-Lumped Fluid

Networks", *Trans. ASME J. of Dynamic Systems Measurement & Control*, Vol 99, pp34-40

SkarbeK-Wazynski C M, 1981, "Hydraulic System Analysis by the Method of Characteristics", *PhD Thesis*, University of Bath, p77, p106.

Steki J S & Davis D C, "Fluid Transmission Lines - Distributed Parameter Models. Parts 1 & 2", *Proc. IMechE*, Vol 200 A4, 1986.

Tomlinson S P, 1987, "The Hydraulic Automatic Simulation Package (HASP): Modelling and Simulation Aspects", *PhD Thesis*, University of Bath, UK.

Trikha A K, 1975, "An Efficient Method For Simulating Frequency Dependant Friction in Transient Liquid Flow". *Trans. ASME, J. of Fluids Eng.*, Vol 97, pp97-104.

Vecchi M P & Kirkpatrick S, 1983, "Global Wiring by Simulated Annealing", *IEEE Trans. on Computer Aided Design*, Vol. CAD-2, p215.

Viersma T J, 1980, "*Analysis, Synthesis and Design of Hydraulic Servo-systems and Pipelines*", Elsevier Science Publication Corp.

APPENDIX 1

LAMINAR ORIFICE SIMULATION EXAMPLE

§A1.1 INTRODUCTION

The purpose of this appendix is to illustrate the results obtained from the TLM solution of the simple circuit example described qualitatively in §3.2 and compare the results with a simple analytical test case derived using lumped parameters.

§A1.2 PIPE-ORIFICE CIRCUIT EXAMPLE

The pipe-orifice circuit is shown in Figure A1.1. The analytical solution to the pressure transient in the volume P_v for a step change in flow Q_s is derived as follows.

For the finite volume (capacitance C):

$$sP_V = \frac{1}{C}(Q_S - Q_R) \qquad \text{A1.1}$$

and for the laminar restrictor of resistance R_L:

$$Q_S = R_L P_V \qquad \text{A1.2}$$

[Note: resistor discharges into reservoir at zero gauge pressure]

Hence by direct substitution:

$$\frac{P_V}{Q_S} = \frac{1/C}{R_L/C + s} \qquad \text{A1.3}$$

For a step in flow of magnitude Q_s into the finite volume at $t=0$:

$$\frac{P_V}{Q_S} = \frac{1/C}{s(R_L/C + s)} \qquad \text{A1.4}$$

From the inverse Laplace transform the pressure in the pipe volume P_V has the following transient response:

$$P_V = \frac{Q_S}{R_L}\left(1 - \exp\left(-\frac{R_L}{C}t\right)\right) \qquad \text{A1.5}$$

For this example the TLM solution is obtained using the capacitive transmission line approximation (§3.2.1). The sequence of numerical operations for this simple example is as follows (noting that steps 3 and 4 can be interchanged, or computed simultaneously):

STEP 1. Calculate line impedance from capacitance and time step: $Z = \Delta/C \quad \left[C = \frac{V}{B_e}\right]$

STEP 2. Set initial characteristic pressures at line ends: $C_S = 0$ $C_R = 0$

STEP 3. Determine conditions at flow source end of transmission line using characteristic pressure propagated from laminar restrictor:

Flow: Q_S

Pressure: $P_s(t) = C_R(t-T) + ZQ_s(t)$

New characteristic pressure: $C_s(t) = P_s(t) + ZQ_s(t)$

STEP 4. Determine conditions at laminar resistor end of transmission line using characteristic pressure propagated from the flow source:

Pressure: $P_R(t) = \dfrac{C_L(t-T)}{1+ZR_L}$

Flow: $Q_R(t) = R_L P_R(t)$

New characteristic pressure: $C_R(t) = P_R(t) + Z(-Q_R(t))$

STEP 5. Propagate new flow source characteristic pressure to laminar restrictor and propagate new laminar restrictor characteristic pressure to flow source

STEP 6. Increment time by time step: Δ

STEP 7. Repeat steps 2 to 6 (until simulation complete)

Note that the above procedure can be easily programmed into a personal computer or programmable calculator.

Using the following test data the TLM solution for pipe pressure is plotted against the analytical result in Figure A1.2 for different fixed TLM time steps.

Flow source	$\mathbf{Q_S}$	10 [l/min]
Resistance	$\mathbf{R_L}$	1 [l/min/bar]
Pipe capacitance	**C**	5.5×10^{-7} [m^3/bar]

Figure A1.2 demonstrates clearly that the smaller the TLM time-step, the closer TLM becomes to the lumped parameter analytical solution. The TLM solution with time-steps less than about 1ms gave results indistinguishable from the analytical result.

§A1.3 SUMMARY

A simple simulation capable of analytical solution has been computed using fixed step transmission line modelling. This example demonstrates that even with quite large time steps (10ms) the transient response computed using TLM is close to that given by the lumped-parameter analytical solution.

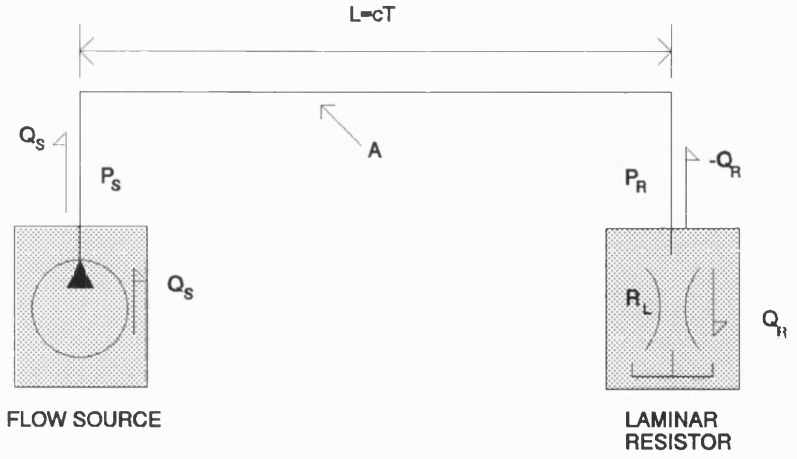


FIGURE A1.1 PIPE-ORIFICE EXAMPLE

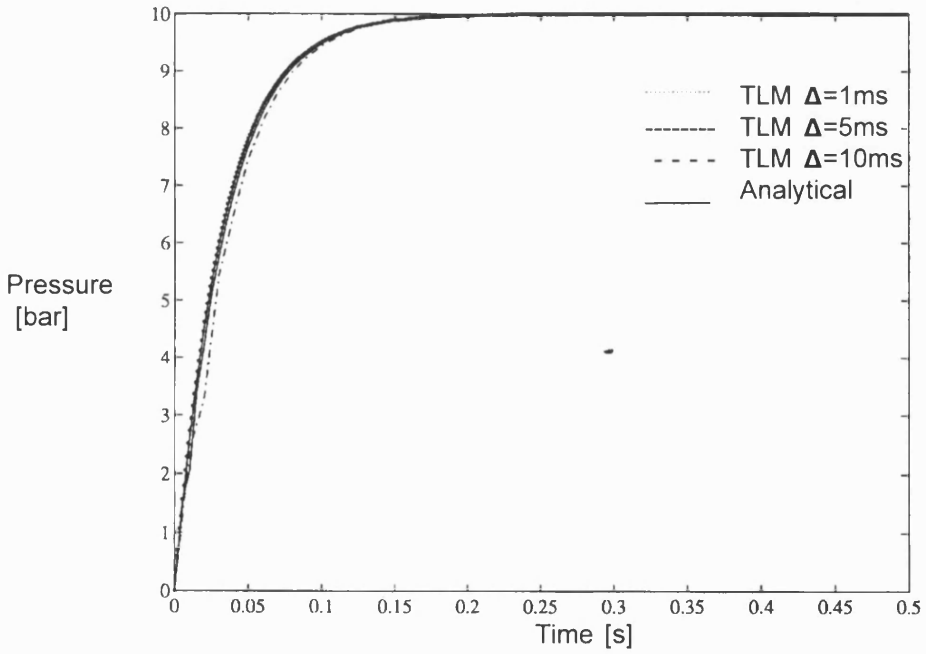


FIGURE A1.2 PRESSURE TRANSIENT FOLLOWING A STEP FLOW INPUT
(TLM vs ANALYTICAL RESULT)

THE AMERICAN SOCIETY OF MECHANICAL ENGINEERS
345 E. 47 St., New York, N.Y. 10017

92-WA/FPST-11

Modelling Requirements for the Parallel Simulation of Hydraulic Systems

J. D. BURTON
K. A. EDGE
C. R. Burrows
Fluid Power Centre
University of Bath
Bath, UK

ABSTRACT

Parallel simulation of systems offers the benefit of increased speed of execution, but requires the system model to be partitioned to enable numerical tasks to be performed concurrently. Hydraulic systems are characterised by a transport delay in the pipelines connecting physical components, which is due to the propagation of waves at the speed of sound through the fluid medium. The transmission delay allows component models to be decoupled for the current time step, enabling a parallel solution; the inputs to each component model are delayed outputs from connected models.

This paper describes a simulation environment suitable for the simulation of hydraulic system performance, using the transmission line modelling approach for the pipelines, decoupling the component models in a hydraulic circuit simulation. Computationally efficient models for cavitation and friction are developed and evaluated. In addition, partitioning strategies for parallel operation are outlined, although these have yet to be implemented.

NOTATION

A	Pipe cross-sectional area
A_o	Orifice cross-sectional area
B_e	Effective fluid bulk modulus (including pipe wall compliance and/or dissolved air)
C	Characteristic pressure, $P+ZQ$
C_d	Orifice flow coefficient
c	Acoustic velocity, $\sqrt{B_e/\rho}$
C_f	Hydraulic motor coulomb friction torque coefficient
C_L	Load viscous damping coefficient
C_s	Hydraulic motor slip loss coefficient
C_v	Hydraulic motor viscous damping coefficient
D_m	Hydraulic motor displacement
d	Pipe diameter
f	Function
I	Inertance
I_1	Modified Bessel function of order 1
K	Orifice constant
L	Pipe length
m	Weighting function constant - dimensional form
$\bar{m}$	Weighting function constant - dimensionless form
n	Integer
$\bar{n}$	Weighting function exponent - dimensionless form
P	Pressure
P_t	Reservoir (tank) pressure
P_{cav}	Saturated vapour pressure
P_J	Junction pressure
Q	Flow
Q_c	Component flow
Q_l	Line flow
Q_m	Hydraulic motor flow
Q_o	Orifice flow
Q_t	Reservoir (tank) flow
R	Resistance coefficient, $\partial^2(\Delta P)/\partial Q \partial x$ ($8\mu/\pi r^4$ for laminar friction)
r	Pipe radius
s	Laplace operator
T	Wave propagation time
T_L	Load torque
T_R	Resistive torque
t	Time, L/c
V	Pipe volume
V_{cav}	Vapour cavity volume
x	Longitudinal pipe coordinate
Y	Integral
y	Recursive term in weighting function
Z	Line characteristic impedance
Z_J	Effective junction impedance
z	z-transform operator, $z = e^{s\Delta}$
Δ	Global time step
γ	Propagation operator
κ	Attenuation factor
λ	Convolution variable
μ	Dynamic viscosity
ν	Kinematic viscosity
ρ	Fluid density
τ	Non-dimensional delayed time, $(t - T)/\tau_0\ T$
τ_0	Non-dimensional delay time, $\nu T/r^2$
ω	Viscous transmission line impulse response (non-plane waves)
$\tilde{\omega}$	Approximate weighting function
ω_m	Hydraulic motor angular speed

Presented at the Winter Annual Meeting
November 8-13, 1992—Anaheim, CA

INTRODUCTION

Numerical simulation of hydraulic systems is used extensively as a design tool, but in order to investigate much larger and more complex systems, within an economic time frame, faster software and hardware is required. A reduction in execution time will improve the conventional uses of simulation and will provide a potential real-time capability. For the latter case, non-linear model reference adaptive control, sophisticated on-line condition monitoring and human interaction with simulated plant all become possible application areas.

Lumped Parameter Modelling

Realistic modelling of hydraulic plant frequently demands a large number of state variables, if all dynamic elements are represented as first order differential equations. These systems are characterized by numerical difficulties, such as strong non-linearities and highly coupled, stiff, differential equations. In addition, the numerical models may be discontinuous, with different modes of operation modelled by different continuous equations [Richards et al, 1990]. As a consequence the numerical algorithms necessary to deal with the stiffness problem have to locate each discontinuity point and restart the integrator at this point, limiting the simulation speed.

As a practical example consider the hydraulic circuit in Figure 1, which shows an application involving the synchronized operation of two actuators, simulated at Bath University Fluid Power Centre. Each group of inter-connected lines are combined to form a lumped volume, compressibility-only line model, causing all components attached to the volume (pumps, valves, actuators etc.) to be instantaneously connected and therefore very closely coupled. Coupling between component models is dependent upon the ratio of bulk modulus to volume in the lumped line model; very small volumes result in very fast transients. The requirements of an efficient numerical integration algorithm for such a system are therefore very demanding, particularly for large systems of state variables. The computer simulation of Figure 1 contains twenty-nine models, most of which include discontinuities and non-linear elements, incorporating eleven state variables. Depending upon the actuator loads specified, a simulation time of twenty seconds required approximately thirty-five minutes processing time, running on a Sun3 68020 micro-processor, considerably slower than real-time.

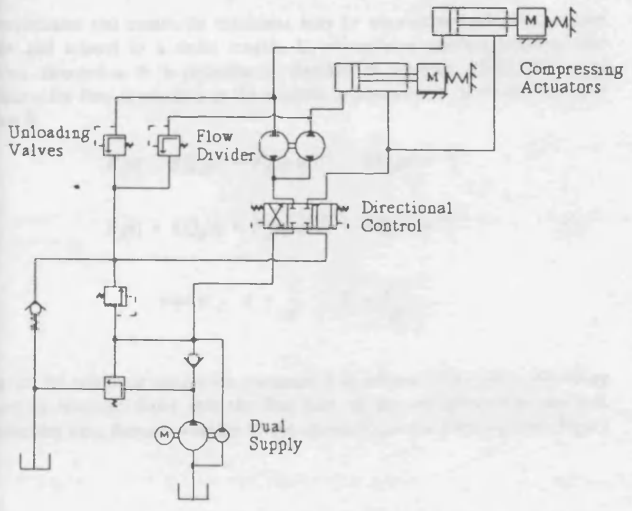


Figure 1 Synchronised Actuator Hydraulic Circuit

Distributed Parameter Modelling

Real-time simulation of such a circuit is only feasible with substantially increased processor speed, or by partitioning the simulation into parallel tasks on separate, communicating processors. Parallel operation requires a convenient method for dividing a hydraulic system into separate numerical tasks. Partitioning of the lumped numerical analysis, as mentioned above, is non-trivial and highly system dependent. Links of *weak coupling* (large volume lumped lines, for example) could provide convenient points at which to divide a simulation. However, the Jacobian matrix describing the system is likely to change significantly with strong non-linearities and at discontinuity points, which in turn may change the coupling between components from weak to strong.

A study of a variety of complex circuits revealed that most processing time is actually spent performing centralised integration, as opposed to computing code specific to each component model. In the case of the circuit shown in Figure 1, for example, numerical tests revealed that approximately sixty percent of CPU time was used for integration alone. As the integrator (described in Richards et al [1990]) is itself not amenable to parallelization, due to its sequential design, only the code relating to the components is amenable to a parallel implementation; the resultant gain in processing speed would be quite small.

An alternative approach is suggested by Krus et al [1990], which utilises the transport delay in the pipelines that connect components together. If the transmission of information is restricted to the speed of wave propagation, then there is no immediate communication of information between components connected by *distributed parameter* line models. Consequently there is no requirement to solve a large, monolithic system of coupled differential equations at every discrete time step, as each component model is decoupled from its neighbours.

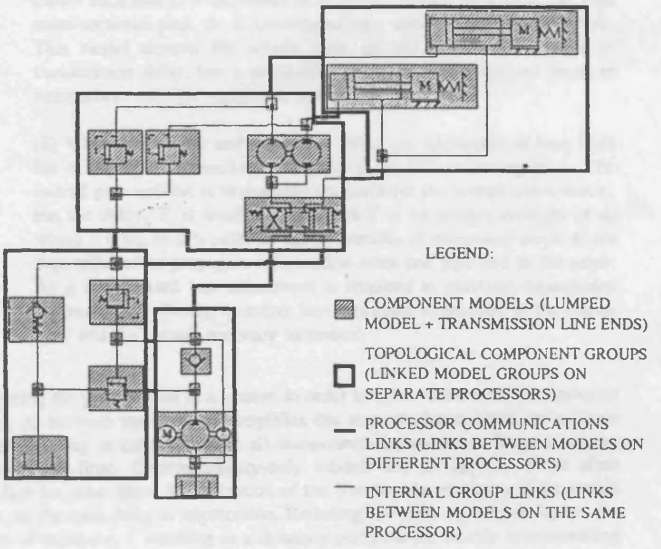


Figure 2 Hydraulic Circuit Partitioned into Topological Groups

Two alternatives are possible if wave propagation is to be incorporated into the line model:

(i) **The method of characteristics** (MOC), as described by Fox [1977]. With this method each line is discretised into a series of internal points at the intersection of *characteristic lines*; pressure and flow velocity is then calculated at each of these internal points. Wazynski [1981] investigated the application of this approach to general fluid power systems analysis, but found the method of characteristics was not able to model small trapped volumes (eg. manifolds and fluid volumes in valves etc.) as accurately as the lumped-parameter, compressibility-only model. A combination of MOC and lumped-parameter simulation, with centralised integrator control, was considered most appropriate.

(ii) **Transmission line modelling** (TLM). In this case the line is represented as a two port, four terminal transmission line and the pressures and flows are only evaluated at each end of the line [Karam 1972] [Boucher 1986] [Krus et al 1990]. Transmission line methods have been applied to electrical and mechanical, as well as fluid power systems [Johns & O'Brien 1980] [Kitsios & Boucher, 1896] [Partridge, Christopoulis & Johns, 1987]. Consequently, plant containing all these elements, such as an electro-hydraulic servo control system, may be modelled using the same numerical techniques for the hydraulic, mechanical and electrical parts.

In general, TLM, requires less computational effort than the method of characteristics, as no internal points are calculated. In addition, the component models forming a system are simplified, as the equations relating pressure and flow at the transmission line ends are incorporated into the model. The total number of models is then reduced, as the line models are not solved separately; in MOC each line must be solved first using the boundary conditions provided by connecting models.
Figure 2 shows the previous example; now the pipes are not combined into lumped volumes, but treated as individual transmission line delay elements, connected by line junctions and component models. Computer models now only exist for each component and line junctions. Groups of connected components (topological groups) may then be assigned to individual processors, four in the example shown in the figure.

HYDRAULIC LINE MODELLING

In this section computationally efficient and accurate transmission line models for pipelines and fluid volumes are developed, which enable the circuit components of a computer simulation to be numerically decoupled and therefore easily partitioned into parallel tasks.
Consideration of a *lossless* fluid element in a semi-rigid pipe of cross sectional area A, enables the derivation of partial differential equations for momentum and continuity at a point [Streeter, 1961].

Continuity Equation:

$$-\frac{\partial Q}{\partial x} = \frac{A}{B_e}\frac{\partial P}{\partial t} \qquad (1)$$

Momentum Equation:

$$-\frac{\partial P}{\partial x} = \frac{\rho}{A}\frac{\partial Q}{\partial t} \qquad (2)$$

The momentum and continuity equations may be transformed into the Laplace domain and related to a finite length, L, of uniform conduit between two locations, denoted a & b [Blackburn, Reethof & Sheerer, 1960]. The sign convention for flow is positive in the positive x-direction, ie. from position a to position b.

$$P_a(s) - ZQ_a(s) = P_b(s)\,e^{-\gamma L} - ZQ_b(s)\,e^{-\gamma L}$$

$$P_b(s) + ZQ_b(s) = P_a(s)\,e^{-\gamma L} + ZQ_a(s)\,e^{-\gamma L} \qquad (3)$$

$$\text{where} \quad \gamma = \frac{s}{c}, \quad Z = \frac{\rho c}{A}$$

However, for computer simulation purposes it is advantageous when modelling systems to consider flows into the line (out of the component) as positive. Consider the time domain solution to the *symmetrical* transmission line, Figure 3.

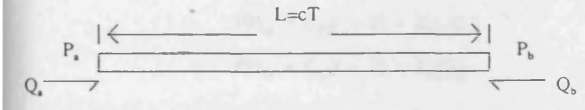


Figure 3 Symmetrical Transmission Line

The symmetry of this configuration results in identical forms for the *characteristic* equations relating each end of a lossless transmission line, eq.(4). Identical forms of the equations describing the line allows the equation for each line termination to be combined with a component model and retain the same sign convention; flows from the model are positive. This enables component models to be connected to form a system without further regard to sign conventions for flow.

$$P_a(t) - ZQ_a(t) = P_b(t-T) + ZQ_b(t-T) = C_b(t-T)$$

$$P_b(t) - ZQ_b(t) = P_a(t-T) + ZQ_a(t-T) = C_1(t-T) \qquad (4)$$

The right hand sides of eq.(4) are referred to as characteristic pressures, representing delayed information from the opposite pipe end. The transmission line is represented as a four-port terminal network, with four possible configurations (pressure or flow input or output at either line end). An entire system is now represented by a series of component models (reservoirs, valves, pumps, actuators etc.) inter-connected via distributed parameter transmission lines.

Based on the above methodology, two alternatives exist for modelling pipes as transmission lines:

(i) Consider the line to be predominantly capacitive (V/B_e constant) [Krus et al 1990]. In this case the pipe length, L, is adjusted in the model such that $L = c\Delta$, where Δ is the simulation time step; the pipe cross-sectional area, A, is correspondingly altered such that $A = V/L$. This model distorts the inertia term ($\rho L/A$) due to the change in transmission delay, but is acceptable provided that it remains small in comparison with the capacitive term.

(ii) Where the inertia and capacitive terms *are* important, in long lines for example, it is possible to discretize the line according to Δ. The overall pipe volume is maintained (to maintain the overall capacitance), but the delay, T, is modified such that T is an integer multiple of Δ, where $T = n\Delta$. In this case, an integer number of integration steps, n, are then required to propagate information from one pipe end to the other. As n is increased less adjustment is required to maintain capacitance (overall pipe volume), therefore less distortion is required in the inertia term and the overall accuracy increases.

Distorting the pipe lengths in a system in order to achieve the same transmission delay, Δ, between components simplifies the numerical procedure, as a single fixed step may be used to isolate all components connected to lumped volume transmission lines. Compressibility-only models are an approximation often justified for short lines; discretization of the line is only necessary if the inertia term, or the time delay is appreciable. Reducing the time Δ, reduces further the effect of inertance, I, resulting in a dynamic performance closely approximating that of a purely capacitive line. Pipeline inertance is expressed in terms of the pipe delay, Δ, and volume, as follows:

$$I = \frac{\rho L}{A} = \frac{\rho L^2}{V}, \qquad L = c\Delta$$

$$\therefore \; I = \frac{\rho c^2 \Delta^2}{V} \qquad (5)$$

COMPONENT MODELLING

Components may be linked by the characteristic equations to form a complete circuit description. Each component model must provide either pressure or flow to enable the other variable to be determined explicitly from the pipeline characteristic equation. There always exists a relationship between the characteristic equation and the equation describing the component at each port of the model:

$$\text{pipe} \qquad P_{a,b}(t) - ZQ_{a,b}(t) = C_{b,a}(t-T)$$

$$\text{component} \qquad P_{a,b} = f_1(Q_{a,b}) \quad \text{or} \quad Q_{a,b} = f_2(P_{a,b}) \qquad (6)$$

Pressure Source

In order to explain how a component model is derived, firstly consider a simple component, such as a pressure source, Figure 4. In this case the constant reservoir pressure, P_1, is substituted directly into the line characteristic equation to obtain

the flow, Q_t, out of the component, eq.(7). The first suffix of the line pressures and flows indicate the model port number (only one port in this model) and the second, the transmission line end.

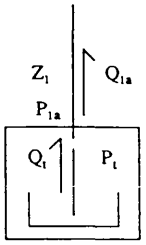


Figure 4 Constant Pressure Source Model

$$\textit{pipe} \quad P_{1a}(t) - Z_1 Q_{1a}(t) = C_{1b}(t - T)$$

$$\textit{component} \quad P_{1a} = P_t \quad \& \quad Q_{1a} = Q_t \qquad (7)$$

$$\therefore \quad Q_t = \frac{1}{Z_1}(P_t - C_{1b}(t - T))$$

Orifice Model

Now consider the turbulent orifice shown in Figure 5, as an example of a non-linear passive component.

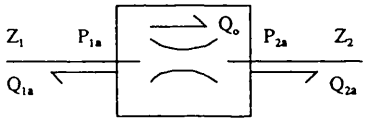


Figure 5 Transmission Line Orifice Model

The internal sign convention for flow within the orifice is from port 1 to port 2. The orifice flow is related to the pipe flows by:

$$Q_o = -Q_{1a} = Q_{2a} \qquad (8)$$

Pressure differential and flow through the orifice are related by a square law relationship:

$$Q_o = C_Q A_o \sqrt{\frac{2(P_{1a} - P_{2a})}{\rho}} \qquad (9)$$

Flow is determined explicitly by substitution of the pipe-end (characteristic) equations into the component (orifice) equation.

When flow is in the positive direction (port 1 to port 2) $C_{1a} > C_{2a}$ the orifice flow

$$\begin{aligned} P(t)_{1a} &= C_{1b}(t - T) + Z_1(-Q_o) \\ P(t)_{2a} &= C_{2b}(t - T) + Z_2(Q_o) \end{aligned} \qquad (10)$$

obtained from the solution of the resultant quadratic equation:

$$Q_o = -\frac{K^2}{2}(Z_1 - Z_2) + \frac{1}{2}\sqrt{K^4(Z_1 + Z_2)^2 + 4K^2(C_{1b} - C_{2b})} \qquad (11)$$

$$\textit{where} \quad K = C_Q A_o \sqrt{\frac{2}{\rho}}$$

The positive quadratic root is appropriate, as the flow, Q_o, is zero when the differential pressure, C_{1b}-C_{2b}, is zero.

Multi-port Junction Model

In order to construct more complex circuits, where many components are connected to the same fluid volume, a multi-port line junction is required, as in Figure 6. The boundary conditions to this idealized model are that pressure at each port of the node is the same and the flows into the node satisfy continuity.

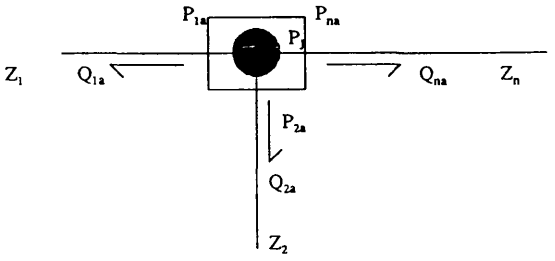


Figure 6 Transmission line Junction Model

For the junction:

$$\sum_{i=1}^{n} Q_{ia}(t) = 0 \qquad P_J(t) = P_{1a}(t) = \ldots P_{na}(t) \qquad (12)$$

The characteristic equation for each pipe is:

$$P_{ia}(t) - Z_i Q_{ia}(t) = C_{ib}(t - T) \qquad (13)$$

Algebraic manipulation of eq.(12) and eq.(13) results in the following explicit equation for junction pressure:

$$P_J = Z_J \sum_{i=1}^{n} \frac{C_{ib}(t - T)}{Z_i} \qquad \frac{1}{Z_J} = \sum_{i=1}^{n} \frac{1}{Z_i} \qquad (14)$$

Where Z_J is the effective *junction impedance*. Flows at the junction are determined from eq.(15) and enable characteristic information ($P_J + Z_i Q_J$) from the junction to be propagated back to connecting models at the next time step.

$$Q_{ia} = \frac{1}{Z_i}(P_J - C_{ib}(t-T)) \qquad (15)$$

Hydromechanical Components

For components that involve differential equations, fixed step integration is necessary, as fixed steps are imposed by the connecting fluid transmission lines. An effective means of obtaining a digital form for such a component is z-transformation, or as a good approximation bi-linear transformation, of the model transfer function. Bi-linear transformation is equivalent to trapezoidal integration of the component differential equation and such a scheme is outlined in Krus et al [1990]. Wazynski [1981] simply uses fixed-step Euler integration for components connected by method of characteristics lines, although 4th order Runge-Kutta is a more common method. If a distributed model of the load is required for increased accuracy, the load system (stiffness and inertia) may be considered to be distributed and therefore governed by wave equations, analogous to those in the fluid lines [Partridge, Christopoulis & Johns 1987].

As an example, consider the hydraulic motor with linearized loss coefficients [McCandlish & Dorey, 1983] and combined inertial load of Figure 7. The characteristic line equations at both ports, the load transfer function and the hydraulic motor equations are given in eq.(16) through to eq.(20).

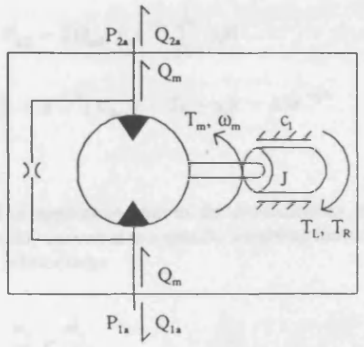


Figure 7 Hydraulic Motor-load Model

Hydraulic motor ideal flow plus leakage flow:

$$Q_m = D_m \omega_m + \frac{c_s D_m}{\mu}(P_{1a} - P_{2a}) \tag{16}$$

Pipe characteristic equations:

$$P_{1a} = C_{1a} e^{-sT} + Z_1 Q_{1a}$$

$$P_{2a} = C_{2b} e^{-sT} + Z_2 Q_{2a} \tag{17}$$

Motor flow to pipe flow:

$$Q_m = Q_{2a} = -Q_{1a} \tag{18}$$

Motor ideal torque less coulomb and viscous torque:

$$T_m = D_m(1 - c_f)(P_{1a} - P_{2a}) - c_v D_m \mu \omega_m \tag{19}$$

Load transfer equation:

$$Js\omega_m = T_m - (T_L + T_R) - c_t \omega_m \tag{20}$$

Combining eq.(16) through eq.(20), to obtain an overall transfer function for motor flow, Q_m:

$$Q_m = \frac{\left(D_m^2(1-c_f) + \frac{c_s D_m}{\mu}(Js + c_t + c_v D_m \mu)\right)(C_{1b} - C_{2b}) e^{-sT} - D_m(T_L + T_R)}{\left(\frac{c_s D_m}{\mu}(Js + c_t + c_v D_m \mu) + D_m^2(1-c_f)\right)(Z_1 + Z_2) + (Js + c_t + c_v D_m \mu)} \tag{21}$$

An explicit form of eq.(21) for digital computation may be obtained using the bi-linear transform, $s = \frac{2}{\Delta}\left(\frac{1 - z^{-1}}{1 + z^{-1}}\right)$

FRICTION & CAVITATION

Utilizing transmission line methods for general hydraulic simulation depends upon successful modelling of friction and cavitation. These models must be computationally efficient (if real-time operation is to be achieved), as well as being sufficiently accurate for the majority of engineering problems.

Distributed Friction

Modelling distributed laminar friction accurately requires the solution of characteristic equations such as in eq.(22); these are the exact analogy to the electrical transmission line problem analyzed by Carslaw & Jaeger [1963].

$$P_{a,b} - ZQ_{a,b} = \kappa C_{b,a}(t-T) + \frac{\kappa TAR}{2\rho}\int_T^t C_{b,a}(t-\lambda) e^{\frac{-AR}{2\rho}\lambda} \frac{I_1\left(\frac{AR}{2\rho}(\lambda^2 - T^2)\right)^{\frac{1}{2}}}{(\lambda^2 - T^2)^{\frac{1}{2}}} d\lambda$$

$$\text{where} \quad \kappa = e^{\left(\frac{ART}{2\rho}\right)} \tag{22}$$

If only steady-state pressure loss associated with the line is important, then the integral term on the RHS of eq.(22) may be neglected. Laminar (or turbulent) friction is incorporated into the attenuation factor κ, which acts to reduce the characteristic pressures at each wave reflection. Krus, Weddfelt & Palmberg [1991] suggest a good approximation to eq.(22), which models distributed friction quite closely.

The distributed approach gives accurate steady-state losses, but ignores frequency-dependent friction due to radial effects (non-plane waves) [Brown 1962] [Trikha 1975]. In simulation, the attenuation of high frequencies is desirable, as distributed friction often fails to attenuate waves realistically, or quickly enough, sometimes resulting in excessive oscillations that are unrepresentative of the real system dynamics.

Frequency-Dependent Friction

The equations relating each end of the transmission line for general time-varying inputs are convolution integrals:

$$P_{a,b} - ZQ_{a,b} = e^{-\tau_0}\int_T^t C_{b,a}(t - \lambda)\omega(\lambda)\, d\lambda \tag{23}$$

The RHS represents pure line attenuation and the convolution of the characteristic pressures with the line impulse response. According to Brown [1962] the short-time approximation of the impulse response of a uniform transmission line is given by the following:

$$\omega = e^{-\tau_0}\frac{1}{\tau_0 T}\frac{1}{2\sqrt{\pi(\tau - \tau_0)^3}}$$

$$\tau = \frac{t - T}{\tau_0 T} \qquad \tau_0 = \frac{\nu}{r^2} T \tag{24}$$

The basic limitation of eq.(24) for a liquid filled line is that the damping number, τ_0, is less than 0.1, which is adequate for the majority of industrial hydraulic applications.

A numerical convolution at each time step would be computationally expensive, in execution time as well as storage of past data points. Hence to avoid evaluating a convolution integral at each time increment, a simple linear transfer function approximation to the impulse response of the line is required to account for frequency-dependent friction. Reformulation of the convolution process into a recursive algorithm reduces the computational overhead; Karam & Leonard [1973], Karam & Tindall [1975] and more recently Krus, Weddfelt & Palmberg [1991] have all attempted this, but all rely on a single first order lag term to model the wave dispersion and are all semi-empirical in approach.

Higher order approximations are possible to improve accuracy. Simplification of eq.(24) is achieved by approximating it to the sum of three decaying exponential terms, eq.(25), whilst maintaining the same integral area as the actual impulse response, equal to unity. The integral area constraint is important, otherwise artificial gains, or losses are introduced into the characteristic equations.

$$\bar{\omega} = \sum_{i=1}^{3} \bar{m}_i e^{-\bar{n}_i \tau} \tag{25}$$

In a similar manner to Trikha [1975] for method of characteristics, a simple recursive form is developed [eq.(26)] that only requires characteristic data from the last time step (APPENDIX 1).

$$P_{a,b} - ZQ_{a,b} = e^{-\tau_0} \sum_{i=1}^{3} y_i(t)$$

$$y_i(t) = \frac{m_i}{n_i}\left(1 - e^{-n_i\Delta}\right) C_{b,a}(t - T) + y_i(t - \Delta)e^{-n_i\Delta} \qquad (26)$$

This approach is general in application, due to the dimensionless form of the approximation, which is easily converted to a specific weighting functions for *any* line, using the following relationships:

$$\frac{m_i}{n_i} = \frac{\bar{m}_i}{\bar{n}_i}, \qquad n_i = \frac{\bar{n}_i}{\tau_0 T_0} \qquad (27)$$

In addition, there are no empirical steps in this approach, only the formation of the approximate, non-dimensional weighting function, which may be refined by the addition of further terms.

Experimental Verification. Experimental verification of the above algorithm was achieved by comparing the computed and measured flows at two separate locations in a rigid pipeline connected to the delivery of an axial piston pump (equipment used for pump impedance measurement by the secondary-source method [BS 6335 Part 1, 1990]), illustrated in Figure 8. The flow ripple Fourier spectrum and the effective fluid bulk modulus were determined experimentally from the measurement of pressure fluctuations at three locations in the line [Johnston, 1988]. The Fourier components were combined to form the flow ripple in the time domain.

The test conditions are summarised as follows:
d = 10mm
B_e = 1.65x10^9 Pa
L_1 = 0.45m
L_2 = 0.7m
ρ = 855 Kg/m^3
ν = 16.2x10^{-6} m^2/s

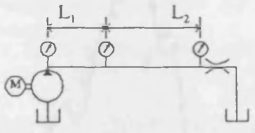
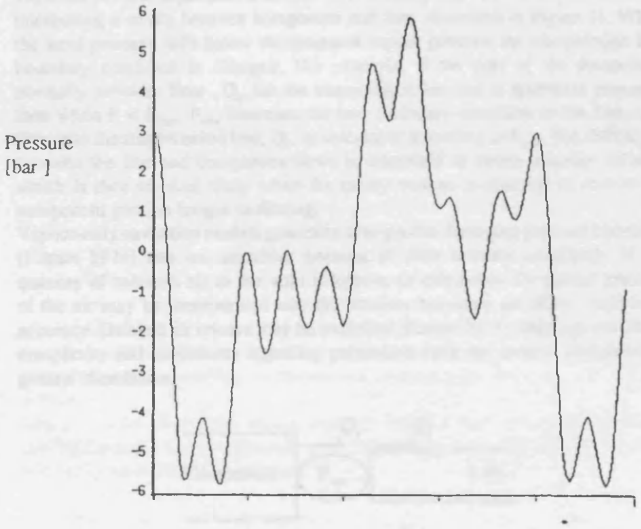


Figure 8 Flow Ripple Experimental Rig

The first ten harmonics of the measured pressure histories at pump discharge and line termination, Figure 9, were used as pressure source inputs to a transmission line model with frequency-dependent friction.
Figure 10 shows the measured and predicted flow at the two locations, in addition to the pressure at an intermediate point, computed by positioning a lossless 2-port junction model at this point in the simulation.The measured and predicted results show very close agreement.

Measured Pressure Ripple at Pump Delivery



Measured Pressure Ripple at Exit

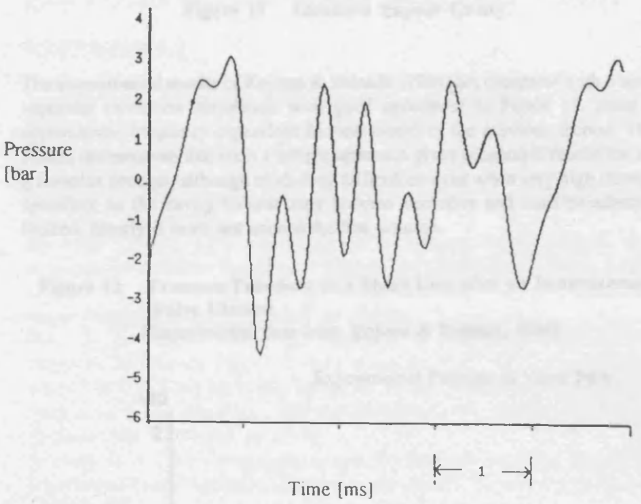


Figure 9 First Ten Harmonics of the Measured Pressure Histories
[Note: All quantities are variations about a mean value]

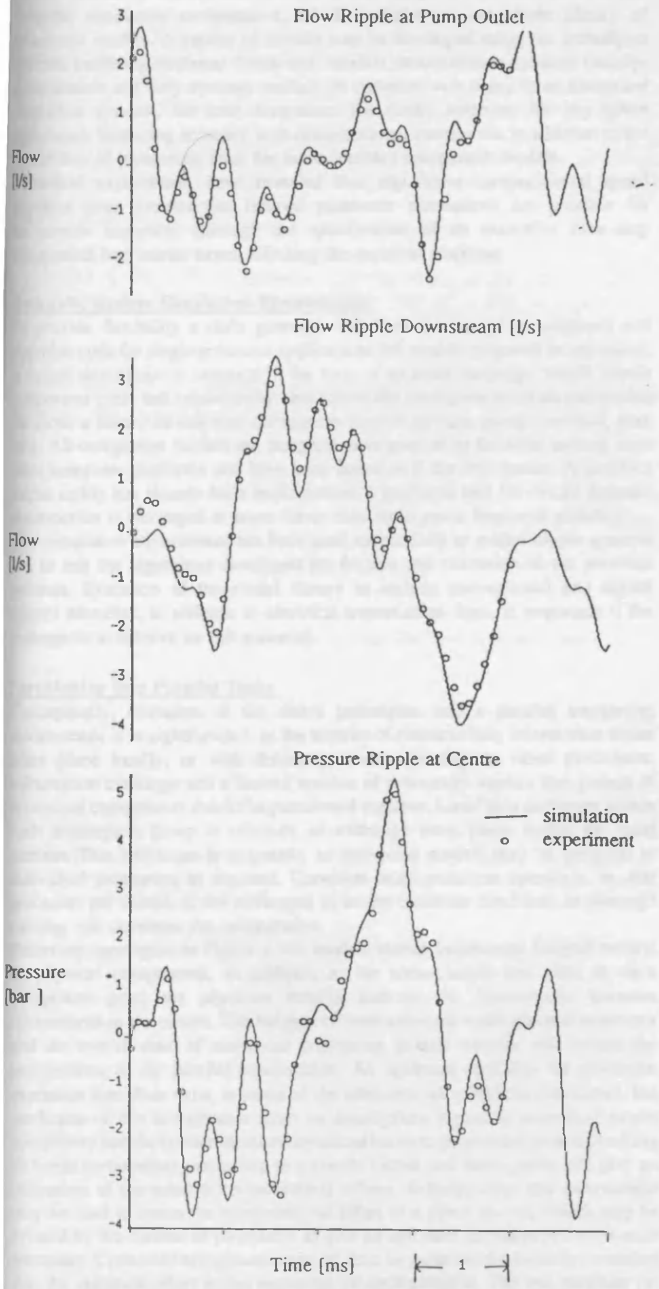
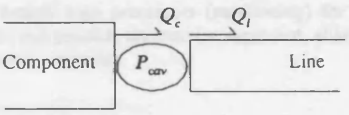


Figure 10 Flow Ripple Measurements Downstream of an Axial Piston Pump During Constant Speed Operation
[Note: All quantities are variations about a mean value]

Cavitation

Detailed modelling of cavitation is highly complex, involving the liberation of dissolved air, and saturated fluid vapour, which may be distributed throughout the line as bubbles, or concentrated in a cavity, or a combination both. To model vaporous column separation a simple void tracking algorithm is implemented by interposing a cavity between component and line, illustrated in Figure 11. When the local pressure falls below the saturated vapour pressure the transmission line boundary condition is changed. For example, if the port of the component normally provides flow , Q_c, for the transmission line end to determine pressure, then when $P < P_{cav}$, P_{cav} becomes the new boundary condition to the line end; flow into the transmission line, Q_l, is calculated according to P_{cav}. The difference between the line and component flows is integrated to obtain a cavity volume, which is then *tracked*. Only when the cavity volume is restored to zero is the component port no longer cavitating.

Vapour-only cavitation models generally over-predict damaging pressure transients [Karam 1974] and are attractive because of their inherent simplicity. If the quantity of released air in the void is known, or calculable, the partial pressure of the air may be incorporated into the pressure boundary condition, improving accuracy. Delayed air release may be modelled [Karam 1974], although the added complexity and uncertainty regarding parameters limit this models usefulness in general simulation.

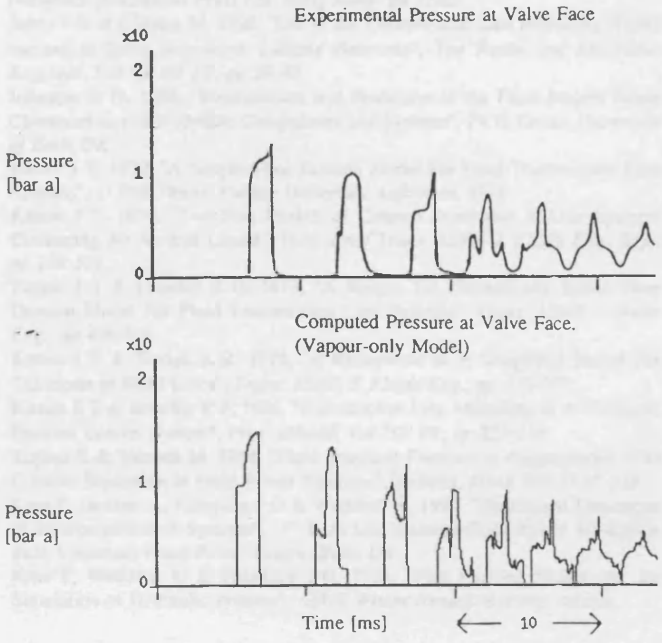


$$V_{cav} = \int_{t_s}^{t_e} (Q_l - Q_c)\, dt$$

Figure 11 Idealized Vapour Cavity

The experimental results of Kojima & Shinada [1984] are compared with a simple vaporous cavitation simulation with good agreement in Figure 12, using the approximate frequency-dependent friction model of the previous section. These results demonstrate that such a simple approach gives acceptable results for such a complex process, although modelling difficulties exist when very high flows are specified, as the cavity volume may become excessive and must be arbitrarily limited; clearly it must not exceed the line volume.

Figure 12 Pressure Transient in a Short Line after an Instantaneous Valve Closure
[Experimental data from Kojima & Shinada, 1984]



NUMERICAL SIMULATION

The development of reliable models for distributed lines, including good approximations for friction and cavitation, has enabled the creation of a general hydraulic simulation environment, which includes an extendible library of component models. A variety of models may be developed using the techniques outlined, including empirical 'black-box' models, instantaneous dynamic (steady-state) models and fully dynamic models. In common with many other numerical simulation systems, the user determines the model selection for any given application, balancing accuracy with computational constraints, in addition to the availability of parametric data. for more complex component models.
Numerical experiments have revealed that significant computational speed increases over conventional lumped parameter simulations are possible for comparable accuracy, although the specification of an excessive time step exaggerates line inertia terms, affecting the transient response.

Hydraulic System Simulation Environment

To provide flexibility a code generator has been developed to construct and assemble code for single processor applications (all models executed in sequence). A circuit description is supplied in the form of an input language, which details component types and connectivity; this allows the configuration of an executable file from a library of run-time component models (orifice, pump, junction, tank etc.). All component models are designed to be general to facilitate porting onto other computer platforms and have been coded in C for this reason. A graphics output utility has already been implemented; a graphical tool for circuit diagram construction is envisaged at some future time to improve front-end usability.
This simulation environment has been used successfully to model simple systems and to test the algorithms developed for friction and cavitation in the previous sections. Extension of the model library to include conventional and digital control elements, in addition to electrical transmission lines, is important if the package is to achieve its full potential.

Partitioning into Parallel Tasks

Conceptually, extension of the above techniques into a parallel computing environment is straightforward, as the transfer of characteristic information either takes place locally, or with distant processes running on other processors. Information exchange and a limited number of processors implies that groups of connected components should be partitioned together. Local data exchange within each topological group is efficient, as exchange takes place within the local process. This technique is adaptable, as individual models may be assigned to individual processors as required. Complete multi-processor operation, ie. one processor per model, is not envisaged to be the optimum condition, as message passing will dominate the computation.
Referring once again to Figure 2, the models shown incorporate lumped models of physical components, in addition to the transmission line ends at each component port; the pipelines merely indicate the connectivity between components in the circuit. The balance of computational loads on each processor and the overall ratio of numerical processing to data transfer will dictate the performance of the parallel computation. An optimum condition for minimum execution time does exist, in terms of the allocation of models to processors, but prediction of this arrangement relies on assumptions regarding individual model complexity and the communications overhead between processors. Bench-marking different components (according to a simple circuit and duty-cycle) will give an indication of the relative computational efforts. Subsequently, this information may be used to assess the computational effort of a given system, which may be divided by the number of processors to give an optimum (or ideal) effort for each processor. Connected components may, at first, be partitioned arbitrarily, ensuring that the optimum effort is just exceeded for each partition. The last partition (or group) to be assigned components will, of course, will have less than the optimum processing effort. Initially a function to be optimized must be chosen and different values of the function obtained by moving connected components into and out of adjacent groups. Such a function, for example, is defined as the sum of the root-mean-square differences between the ideal and actual processing efforts of each processor; the nearest value to zero being the optimum condition. Penalties for inter-processor communication are also included into the function by modifying the processing efforts of those models responsible for data transfer across processors. These partitioning algorithms are similar in principle to those used in optimal PCB design in the electronics industry [Vecchi & Kirkpatrick, 1983].
Such algorithms assume the processing efforts for each model remain constant throughout the simulation, which is not always true; when a model passes through a discontinuity point, for example, the model equations change abruptly and so does the processing effort. A more sophisticated partitioning algorithm may move models onto different processors during the simulation, although this in itself will require additional processor time. The principle aim within the duration of this research is to develop a fast pre-processor, capable of near optimum model partitioning and parallel code generation.

CONCLUSIONS

A technique using transmission line methods (TLM) to model hydraulic pipelines has been developed to allow the convenient partitioning of component models in a hydraulic circuit simulation into parallel processes. Parallel operation is easily accomplished using transmission lines, as the topology of the circuit maps directly onto an array of processors. The propagation delay in the lines modelled using TLM allows components to be numerically decoupled, allowing each model to proceed independently to the next time step; individual model input data is simply the output data of connected components, passed through a filter incorporating the wave delay, attenuation and dispersion of the fluid line. Accurate, but computationally efficient models for simulating unsteady friction and cavitation have been developed using TLM and verified experimentally, demonstrating that transmission line modelling techniques are an appropriate foundation for parallel simulation.
The allocation of models onto processors (partitioning) for maximum speed increase is addressed and possible alternatives suggested, although this research has yet to be evaluated in practice.

ACKNOWLEDGEMENTS

This work is funded jointly by the UK government Science and Engineering Research Council, SERC grant GR/F62971, and the Ministry of Defence.

REFERENCES

Boucher R F & Kitsios E E, 1986, "Simulation of Fluid Network Dynamics by Transmission Line Modelling", *Proc. IMechE, Vol 200, No C1., pp 21-29.*
Blackburn J F, Reethof G & Sheerer J L, 1960, "Fluid Power Control", *MIT Press, p138.*
Brown, F T, 1962, "The Transient Response of Fluid Lines", *Trans. ASME J. Basic Eng., pp 547-553.*
BS 6335 : Part 1, 1990, "Methods for Determining Pressure Ripple Levels Generated in Hydraulic Fluid Power Systems and Components", *British Standard*
Carslaw H S & Jaeger J C, 1963, "Operational Methods in Applied Mathematics", *Dover Publications inc., New York, pp 184-209*
Fox, J A, 1977, "Hydraulic Analysis of Unsteady Flow in Pipe Networks",*MacMillan Press Ltd, Hong Kong, pp 72-85.*
Johns P B & O'Brien M, 1980, "Use of the Transmission Line Modelling *(TLM) method to Solve Non-linear Lumped Networks", The Radio and Electronic Engineer, Vol 50, No 1/2, pp 59-70.*
Johnston D N, 1988, "Measurement and Prediction of the Fluid Bourne Noise Characteristics of Hydraulic Components and Systems", *Ph D Thesis, University of Bath, UK.*
Karam J T, 1972, "A Simple Time Domain Model For Fluid Transmission Line Systems", *D Phil Thesis, Purdue University, Lafayettte, USA*
Karam J T, 1974, "Two Port Models of Column Separation in Line Systems Containing An Aerated Liquid", *Tech. Brief Trans. ASME J. Fluids Eng., Sept. pp 299-301.*
Karam J T & Leonard R G, 1973, "A Simple Yet Theoretically Based Time Domain Model For Fluid Transmission Line Systems", *Trans. ASME J. Basic Eng., pp 498-504.*
Karam J T & Tindall R R, 1975, "A Refinement of A Simplified Model For Transients in Fluid Lines", *Trans. ASME J. Fluids Eng., pp 376-377.*
Kitsios E E & Boucher R F, 1986, "Transmission Line Modelling of A Hydraulic Position Control System", *Proc. IMechE Vol 200 B4, pp 229-236.*
Kojima E & Shinada M, 1984, "Fluid Transient Phenomena Accompanied With Column Separation in Fluid Power Pipelines", *Bulletin, JSME Vol 27 N° 233*
Krus P, Jansson A, Palmberg J-O & Weddfelt K, 1990, "Distributed Simulation of Hydromechanical Systems", *3rd Bath International Fluid Power Workshop. Bath University Fluid Power Centre, Bath, UK*
Krus P, Weddfelt K & Palmberg J-O, 1991, "Fast Pipeline Models for the Simulation of Hydraulic systems", *ASME Winter Annual Meeting, Atlanta,*

Georgia, USA
McCandlish D & Dorey R E, 1983, "The Mathematical Modelling of Hydraulic Pumps and Motors", *Proc IMechE, Vol 198B, N°10, pp 165-174.*
Partridge G J, Christopoulis C & Johns P B, 1987, "Transmission Line Modelling of Shaft System Dynamics", *Proc. IMechE Vol. 201 C4, pp 271-278.*
Richards C W, Tilley D G, Tomlinson S P & Burrows C R, 1990, "Type Insensitive Integration Codes for the Simulation of Fluid Power Systems", *ASME Winter Annual Meeting, Texas, USA*
Streeter V L, 1961, "Handbook of Fluid Dynamics", *Mc Graw Hill Book Co., Inc., p20-9.*
Trikha A K, 1975, "An Efficient Method For Simulating Frequency Dependant Friction in Transient Liquid Flow". *Trans. ASME, Journal of Fluids Engineering, pp 97-104.*
Vecchi M P & Kirkpatrick S, 1983, "Global Wiring by Simulated Annealing", *IEEE Trans. on Computer Aided Design,* Vol. CAD-2, p215.
Wazynski C M, 1981, "Hydraulic System Analysis by the Method of Characteristics", *PhD Thesis, Bath University, UK, p77, p106.*

APPENDIX 1
RECURSIVE APPROXIMATION TO ANALYTICAL IMPULSE RESPONSE

$$y(t) = \int_{T}^{t} C(t-\lambda)\,\omega(\lambda)\,d\lambda - \int_{0}^{t-T} C(\lambda)\,\omega(t-\lambda)\,d\lambda \tag{28}$$

The approximate impulse response (weighting function) is of the form:

$$\omega(\lambda) = \sum_{i=1}^{f} \omega_i(\lambda) \tag{29}$$

$$\text{where} \quad \omega_i(\lambda) = m_i e^{-n_i(\lambda-T)}$$

Therefore consider each element of the weighting function separately and sum the contribution of each convolution, such that:

$$y(t) = \sum_{i=1}^{f} y_i(t) \tag{30}$$

Where:

$$y_i(t) = \int_{0}^{t-T} C(\lambda)\,\omega_i(t-\lambda)\,d\lambda \tag{31}$$

Each component of the convolution after one propagation delay, T, is:

$$y_i(t+T) = \int_{0}^{t} C(\lambda)\,\omega_i(t+T-\lambda)\,d\lambda \tag{32}$$

After a small time increment, Δ, which is smaller than T:

$$y_i(t+T+\Delta) = \int_{0}^{t+\Delta} C(\lambda)\,\omega_i(t+T+\Delta-\lambda)\,d\lambda - \int_{t}^{t+\Delta} C(\lambda)\,\omega_i(t+T+\Delta-\lambda)\,d\lambda + \int_{0}^{t} C(\lambda)\left[\omega_i(t+T+\Delta-\lambda)-\omega_i(t+T-\lambda)\right]d\lambda + \int_{0}^{t} C(\lambda)\,\omega_i(t+T-\lambda)\,d\lambda \tag{33}$$

Each integral of Eq.(33) is considered separately.
The first term of eq.(33) becomes:

$$\int_{t}^{t+\Delta} C(\lambda)\,\omega(t+T+\Delta-\lambda)\,d\lambda - \int_{T}^{T+\Delta} \omega(\lambda)\,C(t+T+\Delta-\lambda)\,d\lambda$$

$\Delta \to 0, \quad \lambda \to T \quad \therefore$

$$-C(t+T+\Delta-T)\int_{T}^{T+\Delta}\omega(\lambda)\,d\lambda - C(t+\Delta)\,m_i\int_{T}^{T+\Delta} e^{-n_i(\lambda-T)} - C(t+\Delta)\frac{m_i}{n_i}\left[e^{-n_i(\lambda-T)}\right]_{T}^{T+\Delta} - C(t+\Delta)\frac{m_i}{n_i}\left(1-e^{-n_i\Delta}\right) \tag{34}$$

The second term of eq.(33) becomes:

$$\int_{0}^{t} C(\lambda)\left[\omega(t+T+\Delta-\lambda)-(t+T-\lambda)\right]d\lambda = m_i\int_{0}^{t} C(\lambda)\left(e^{-n_i(t+T+\Delta-\lambda-T)}-e^{-n_i(t+T-\lambda-T)}\right)d\lambda - \left(e^{-n_i\Delta}-1\right)m_i\int_{0}^{t} C(\lambda)e^{-n_i(t+T-\lambda-T)}d\lambda - \left(e^{-n_i\Delta}-1\right)y_i(t+T) \tag{35}$$

The third term of eq.(33) becomes:

$$\int_{0}^{t} C(\lambda)\,\omega_i(t+T-\lambda)\,d\lambda = y_i(t+T) \tag{36}$$

Substituting these terms (eq.(34)(35)&(36)) back into eq.(33):

$$y_i(t+T+\Delta) = \frac{m_i}{n_i}\left(1-e^{-n_i\Delta}\right)C(t+\Delta) + y_i(t+T)\left(e^{-n_i\Delta}-1\right) + y_i(t+T) = \frac{m_i}{n_i}\left(1-e^{-n_i\Delta}\right)C(t+\Delta) + y_i(t+T)e^{-n_i\Delta}$$

Let $t = t-T-\Delta$

$$\therefore \quad y_i(t) = \frac{m_i}{n_i}\left(1-e^{-n_i\Delta}\right)C(t-T) + y_i(t-\Delta)e^{-n_i\Delta} \tag{37}$$

The complete approximation is the sum of these recursive elements:

$$y(t) = \sum_{i=1}^{f}\frac{m_i}{n_i}\left(1-e^{-n_i\Delta}\right)C(t-T) + y_i(t-\Delta)e^{-n_i\Delta} \tag{38}$$

THE AMERICAN SOCIETY OF MECHANICAL ENGINEERS
345 E. 47th St., New York, N.Y. 10017

93-WA/FPST-4

PARTITIONED SIMULATION OF HYDRAULIC SYSTEMS USING TRANSMISSION-LINE MODELLING

James D Burton
Kevin A Edge
Clifford R Burrows

Fluid Power Centre
University of Bath
Bath
UK

ABSTRACT

This paper describes research into a distributed processor scheme for time-domain simulation of hydraulic circuits, ultimately for use in real-time applications, involving control and on-line condition monitoring. The approach adopted uses transmission line models of the pipelines as a means of decoupling the system components and implements the concepts developed in the authors' preliminary work [Burton, Edge & Burrows 1992].

To assist this study a simulation program generator has been developed which links models of relevant components contained in a library. The program is executed on a multi-transputer platform. Using this facility, a study has been undertaken to assess the most appropriate number of processors and best circuit partitioning strategy for the case of a particular hydraulic circuit. From this investigation, partitioning guidelines are proposed.

NOMENCLATURE

A	Area
B	Bulk modulus
C	Characteristic pressure, Capacitance
c	Acoustic velocity $\sqrt{B_e/\rho}$, Viscous friction
d	Pipe diameter
F	Force
f	Function
i	Integer
K	Flow-pressure valve constant, Spring rate
L	Pipe length
M	Mass
n	Integer
P	Pressure
Q	Flow
s	Laplace operator, Actuator stroke
T	Wave propagation time, L/c
t	Time
v	Velocity
V	Pipe volume
x	Displacement
Z	Line characteristic impedance $\rho c/A$
Δ	Simulation time step
ρ	Fluid density

Subscripts

a	Line end 'a'
b	Line end 'b'
c	Cracking
e	Effective
f	Coulomb friction
k	Spring
L	Load force
R	Resistive force (opposes motion)
S	Stiction force
v	Valve

Presented at the ASME Winter Annual Meeting
New Orleans, Louisiana — November 28—December 3, 1993

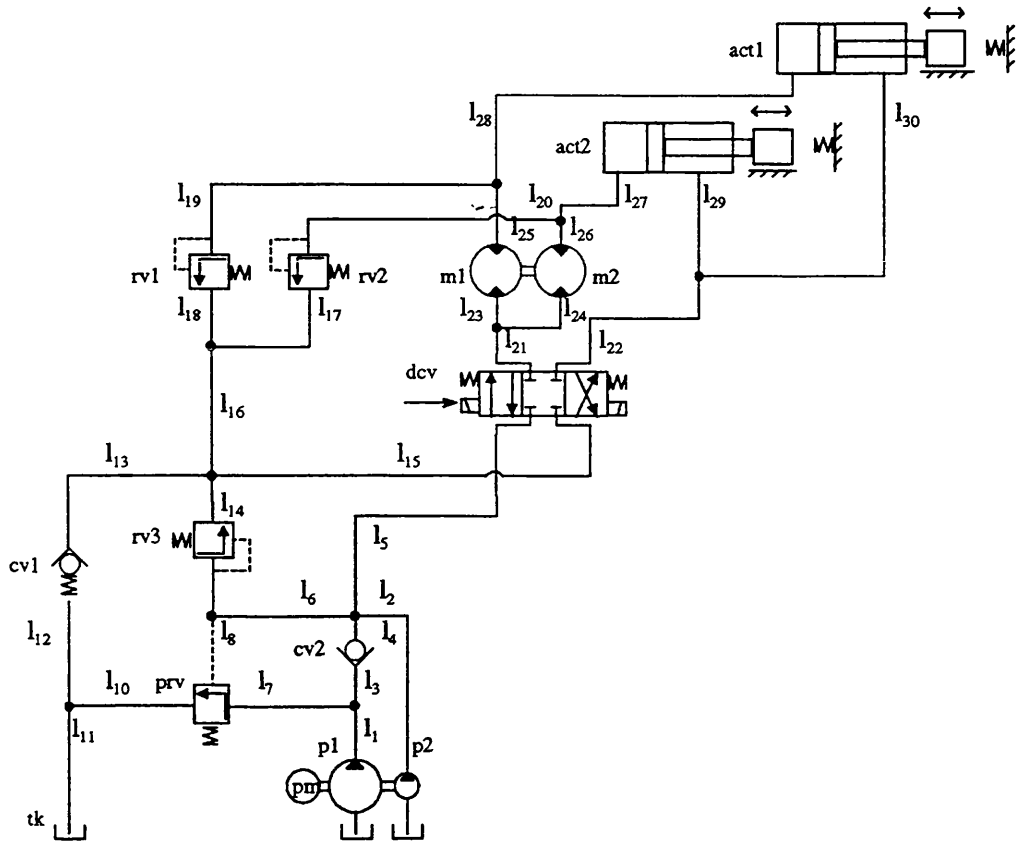


FIGURE 1 ACTUATOR CIRCUIT

INTRODUCTION

The simulation of complex fluid power systems often results in unacceptably long computer execution times, partly because of the close coupling between component models. This coupling may result in numerically-stiff differential equations, which ideally require special numerical integration algorithms for their solution [Richards et al 1990]. An alternative approach is to decouple the component models in a circuit simulation, utilizing the inherent propagation delay modelled by a transmission line. The benefits of this approach, which are well documented [Sidell & Wormley 1977][Johns & O'Brien, 1980][Boucher & Kitsios, 1986][Krus et al 1990], include potentially significant reductions in execution time. However, further reduction in the time for simulation is possible through the use of Transmission Line Modelling (TLM) combined with parallel processing: a system may be divided into a group of connected sub-systems, each simulated on a single processor as part of a multi-processor array.

In this paper a TLM approach to multi-processor simulation of hydraulic systems is presented, with particular reference to the circuit shown in Figure 1. This example circuit, which involves the synchronized operation of two cylinders, is particularly demanding on processor time, when using classical numerical integration methods to solve the highly stiff system differential equations (the ODE solver LSODA [Richards et al, 1990] used over 6300 seconds run time on a Sun Sparc 4/370 workstation, for only 2.5 seconds simulation time). Consequently, it represents an ideal case study with which to examine the potential benefits of a TLM based multi-processor simulation.

This paper outlines the approach to the development of component models for the actuator circuit, shown in Figure 1, and goes on to examine in detail the partitioning of these models onto separate processors to reduce execution time.

THE TLM APPROACH

In order to *illustrate* the approach to modelling, it is convenient to consider a lossless transmission line.

$$P_a - ZQ_a = (P_b + ZQ_b)e^{-sT} = C_b e^{-sT}$$
$$P_b - ZQ_b = (P_a + ZQ_a)e^{-sT} = C_a e^{-sT} \quad (1)$$

where C_a and C_b are defined here as *characteristic pressures* and $Z=\rho c/A$.

The symmetry of this configuration (positive flows into the transmission line) results in identical equations for each end of the line. A TLM component model is developed by combining the component equation (pump, valve, actuator etc.) with the transmission line equation at each port of the component. The symmetry of the line equations ensures that the component models are linked together consistently to form a system model. Inputs to a TLM component model are delayed outputs from connected models (characteristic pressures) and are therefore decoupled in time.

Krus et al [1990] have demonstrated that in order to achieve a global time step for the simulation, it is convenient to "adjust" the lengths of the transmission lines, such that all lines have the same propagation delay, Δ. Provided that compressibility effects are dominant, the line may be considered predominantly capacitive (V/B_e constant). In this case the pipe length, **L**, may be adjusted such that $L=c\Delta$, where Δ is the simulation time step; the pipe cross-sectional area, **A**, is correspondingly altered such that $A=V/L$. This line model distorts the inertia term ($\rho L/A$) due to the change in transmission delay. However, this distortion is acceptable provided that the line inertia contribution remains small in comparison with the capacitive term. Line discretization (sub-division of a line into a series of smaller elements) is only necessary if the time delay is appreciable.

Lossless transmission lines introduce undamped resonances. If unrealistic resonant effects are to be avoided, low pass filtering of the characteristic pressures is necessary, which approximates the frequency-dependent damping evident in actual pipelines. The filtering method described by Krus et al [1990] has been used in the simulation study of the example circuit.

Consideration of all the components connected to transmission lines, requires a *system* of equations to be solved. Using the convention that each component port is connected to a transmission line at end a, we have the component equation:

$$\bar{Q}_a = f(\bar{P}_a) \quad (2)$$

[where the *bar* denotes a vector quantity]
and the transmission-line equations at each component port:

$$\bar{P}_a - \bar{Z}\bar{Q}_a = \bar{C}_b e^{-s\bar{\Delta}} \quad (3)$$

It is then necessary to solve simultaneously the component equation with the transmission-line equation at each component port. The development of models for a range of components and the approach required for their solution, have been dealt with in other publications [Burton et al, 1992, 1993]. Burton et al [1992] also discusses the use of a void tracking algorithm for modelling vaporous cavitation.

Components in Figure 1 which have not been previously considered comprise the directional control valve, the flow divider and the actuators. For completeness, details of how these components have been modelled will now be discussed.

COMPONENT MODELLING

Directional Control Valve Model

The model of a directional control valve is a fairly simple extension of the orifice model, described by Burton et al [1992]. For each valve position a reference pressure drop and corresponding flow are specified, to determine the square-law orifice coefficient for each port combination. Proportional control of the valve may be incorporated by assuming a linear variation in orifice coefficient, equivalent to a linear change in annular flow area across the valve spool.

Flow Divider Model

The flow divider is modelled by extending the motor model [Burton et al, 1992]. If *two* motors are linked by a common shaft then the torques provided by each motor may be solved with the combined equation of motion for the shaft dynamics, resulting in an overall transfer function for shaft speed. An explicit time-difference equation for digital computation of the resultant differential equation is obtained using the bi-linear transform. Substitution of the shaft speed at each time step enables calculation of pressure and flow at each motor port and the respective motor torques.

Relief Valve Model

The relief, or check valve model (Figure 2) assumes that the valve dynamics are instantaneous. For this type of model the valve is uni-directional and no flow across the valve takes place until the differential pressure exceeds the cracking pressure, P_c. A linear flow-pressure characteristic of gradient **K** is assumed once the valve has cracked open. Transmission-line end equations at each valve port, taking account of flow sign:

$$P_{1a} = C_{1b}e^{-s\Delta} - Z_1 Q_v$$
$$P_{2a} = C_{2b}e^{-s\Delta} + Z_2 Q_v \quad (4)$$

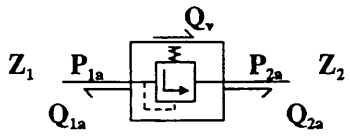


FIGURE 2 RELIEF VALVE MODEL

Equations relating valve flow to differential pressure are:

$$\begin{aligned} (P_{1a}-P_{2a}) > P_c \qquad & Q_v = K(P_{1a}-P_{2a}-P_c) \\ (P_{1a}-P_{2a}) \le P_c \qquad & Q_v = 0 \end{aligned} \tag{5}$$

Flow is determined explicitly by substitution of the pipe-end (characteristic pressure) equations into the valve equation, as follows:

$$P_{1a}-P_{2a} > P_c$$

$$Q_v = K\left(C_{1b}e^{-s\Delta} - C_{2b}e^{-s\Delta} - (Z_1+Z_2)Q_v - P_c\right)$$

$$\therefore \quad Q_v = \frac{K(C_{1b}e^{-s\Delta} - C_{2b}e^{-s\Delta} - P_c)}{1+K(Z_1+Z_2)} \tag{6}$$

$$P_{1a}-P_{2a} \le P_c$$

$$Q_v = 0$$

The valve port pressures and characteristic pressures are then readily determined from Q_v.

Pilot Operated Valves. The above analysis also holds for pilot operated relief and check valve models. However, the valve opening condition is modified to account for pilot operation from a third component port (port 3), instead of the differential pressure across the valve. When the pilot pressure exceeds the valve cracking pressure the valve opens and the calculation proceeds, as before.

The pilot port is modelled as a transmission line with a blocked end at the valve, corresponding to zero flow at the line end. The pilot pressure, therefore, is equal to the characteristic pressure (if $C_{3b}>P_c$ the valve is piloted open). The zero flow condition is a reasonable modelling assumption, as the flow at the pilot port will be negligible.

Differential Area Actuator Model

This model of a differential area actuator includes the integration of two internal state variables (velocity and displacement) and accommodates discontinuities at the end stops, stiction effects, and when the load contacts, or loses contact with a linear spring. Figure 3 shows a schematic of the actuator.

Transmission line end equations at the piston (accounting for the sign notation relating piston velocity to line end flow):

$$\begin{aligned} P_1 &= C_{1b}e^{-s\Delta} - Z_1Q_1 \\ P_2 &= C_{2b}e^{-s\Delta} + Z_2Q_2 \end{aligned} \tag{7}$$

Piston and rod end flows:

$$\begin{aligned} Q_1 &= vA_1 \\ Q_2 &= vA_2 \end{aligned} \tag{8}$$

Piston equation of motion.

Without spring force ($x<x_K$):

$$P_1A_1 - P_2A_2 - F_L - F_R\frac{v}{|v|} - cv = M_e sv \tag{9}$$

Including spring force ($x>=x_K$):

$$P_1A_1 - P_2A_2 - F_L - F_R\frac{v}{|v|} - cv - K(x-x_K) = M_e sv \tag{10}$$

Displacement:

$$v = sx \tag{11}$$

Solving the line, flow and relevant equation of motion, the following equations are obtained.

Without spring force ($x<x_K$):

$$\begin{aligned} & C_{1b}e^{-s\Delta}A_1 - C_{2b}e^{-s\Delta}A_2 - \left(Z_1A_1^2+Z_2A_2^2+c\right)v - \\ & F_L - F_R\frac{v}{|v|} = M_e sv \end{aligned} \tag{12}$$

Including spring force ($x>=x_K$):

$$\begin{aligned} & C_{1b}e^{-s\Delta}A_1 - C_{2b}e^{-s\Delta}A_2 - \left(Z_1A_1^2+Z_2A_2^2+c\right)v - \\ & F_L - F_R\frac{v}{|v|} - K(x-x_K) = M_e sv \end{aligned} \tag{13}$$

The above equations of motion are solved using bi-linear transformation to obtain a pair of discrete time-difference equations (velocity and displacement) for each region of operation.

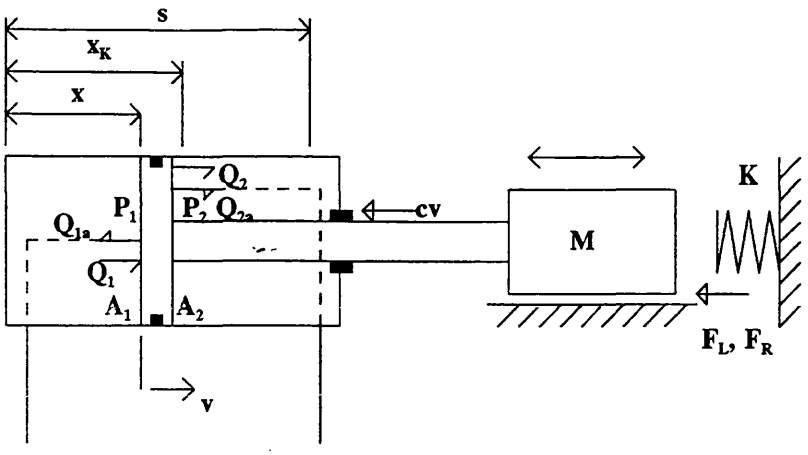


FIGURE 3 TRANSMISSION-LINE ACTUATOR-LOAD MODEL

End Stops. When the actuator displacement from the fully retracted position, x, is less than, or equal to zero, the piston displacement, velocity and acceleration are set to zero. Similarly, when the displacement is greater than, or equal to the stroke limit, s, the displacement is set to the stroke limit and the piston acceleration and velocity set to zero.

Stick-Slip Modelling. Stiction in actuators (due mainly to polymeric piston and rod seals) is modelled using a simple discontinuity region. If the piston speed, v, is within a suitable tolerance band (±1mm/s, say) and the net piston force is less than the static friction (stiction) force, F_S, then the piston is stationary (velocity and acceleration set to zero). If the net piston force exceeds static friction, then motion starts and the static frictional force replaced by a (usually smaller) dynamic friction force, F_R.

Variable Piston and Rod End Volumes. The variable volumes either side of the piston are handled by inclusion with the connecting transmission line volumes, through modification the respective transmission line impedances. This modification of the line impedance has to take place whilst the characteristic pressures are being propagated between component ports and necessitates compensation of the characteristics to maintain the total pressure and total flow in the line. This is also a useful technique for incorporating variations in bulk modulus.

CIRCUIT SIMULATION

Circuit Operation

In this circuit, the coupled motors attempt to divide the flow equally between each actuator, during extension and retraction. The two supply pumps operate together initially, providing maximum flow to the actuators. Once a preset pressure is reached the larger capacity pump is unloaded, via the pilot operated relief valve. The smaller pump continues to supply flow at the higher pressure until either the directional control valve position is altered, or the high pressure relief is activated. This arrangement limits the maximum power required from the prime mover.

One of the drawbacks of this circuit is the inability of the flow divider to supply exactly equal flows. Different leakage rates result in different supplied flows. It is the function of the relief valves connected to the actuators to re-synchronise the actuator positions, at the end of the stroke.

Single Processor Simulation

The salient parameters used in the following simulations are identified on Figure 1 and listed in the Appendix.

A single processor simulation study was first performed to validate the TLM approach. An equivalent *lumped parameter* simulation was also built for this purpose, using the multi-step, variable order stiff numerical integration algorithm LSODA [Richards et al, 1990], with discontinuity handling. It must be noted that the lumped parameter line models used were of the purely capacitive type:

$$sP = \frac{1}{C}\sum_{i=1}^{n} Q_i \qquad C = \frac{V}{B_e} \tag{14}$$

Conversely, the TLM line incorporates a propagation delay equal to the simulation time step and hence a (small) fluid inertance. Nonetheless, this inertia term is a realistic effect. If

Transmission-line Modelling (TLM):

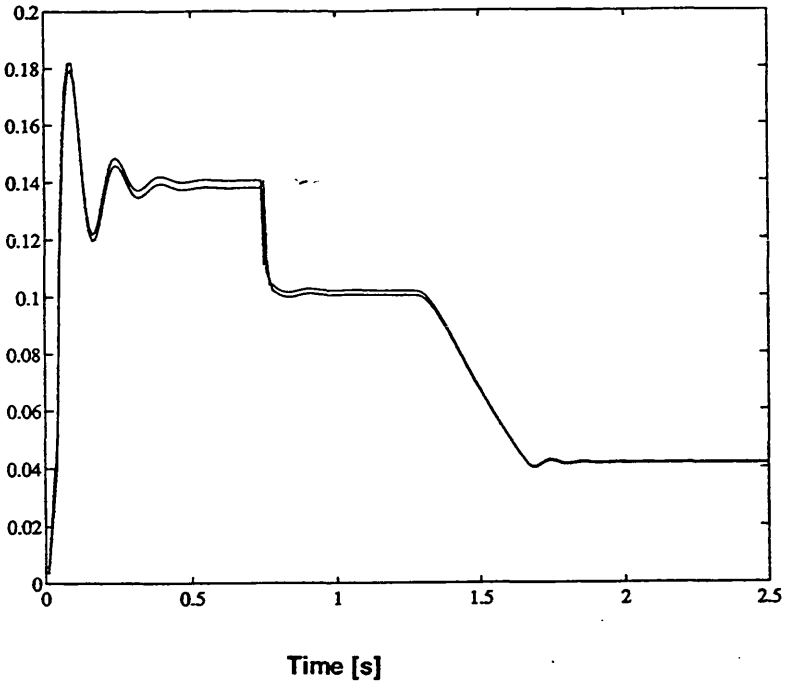


FIGURE 4 ACTUATOR SPEED AGAINST TIME

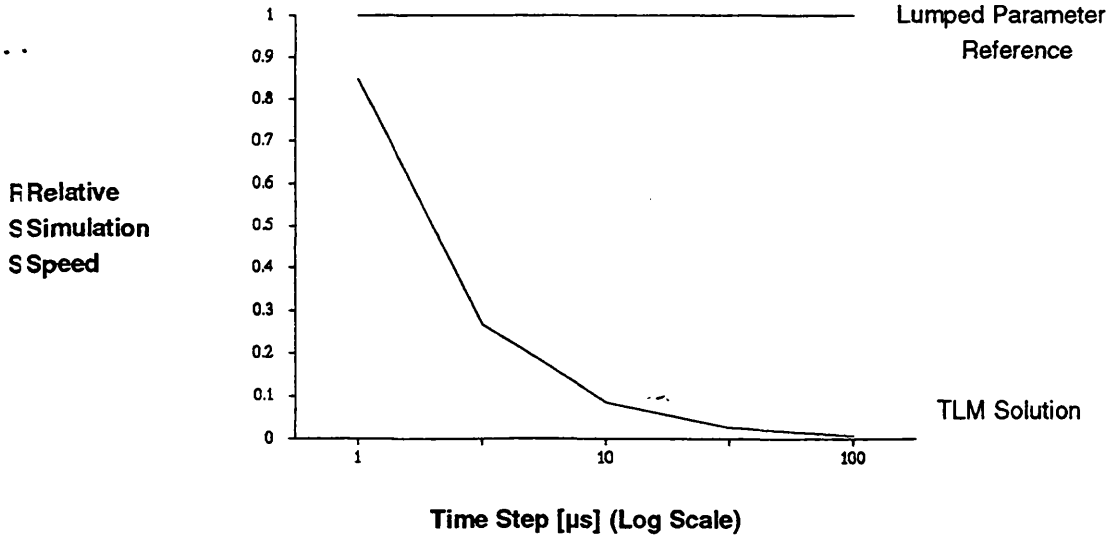


FIGURE 5 NORMALIZED TLM RUN-TIME AS A FUNCTION OF TIME STEP

too large a time step is used, however, the system dynamics may become distorted due to a combination of excessive inertia and "sampling" of the characteristics at too low a rate. Each simulation run involved the prediction of behaviour over a 2.5 second operating cycle, in which the actuators were extending against a spring load.

As an example of the predicted transient behaviour, the velocities of the two actuators are presented in Figure 4. Simulations were repeated for a number of TLM step sizes: only very small differences were noted between the lumped and distributed simulation predictions, even when using a TLM time step of 100μs. Some evidence of wave propagation effects were apparent on close inspection, although these superimposed transients are rapidly damped, because of the low-pass filtering used in the TLM compressible line model.

Figure 5 shows the variation in the single processor TLM simulation speed as a function of the fixed time step used. The transmission line simulations are normalised against the lumped parameter simulation. From the figure, even the 1μs step size requires only 85% of the CPU time required by the lumped parameter equivalent.

DISTRIBUTED SIMULATION

Conceptually, parallel operation of the TLM simulation is straightforward, as the transfer of characteristic pressures between models either takes place locally on the same processor, or with models running on other processors.

With a limited number of processors and the time delay associated with transferring data between them (latency), groups of connected components should be partitioned together. Local data transfer is very efficient, as information is exchanged using shared on-chip memory. The number of processors, the balance of computational loads between processors and the overall ratio of computation-to-communication time will dictate the overall performance of the distributed simulation.

Hardware

The system hardware consists of a **Sun 4/370** host machine, which supports a **4x2 TRAM** (Transputer plus 2 Mbytes RAM) **T800** processor array. Each TRAM incorporates a maximum of four serial data links, to enable data transfer between the processing elements.

Software

GENESYS is a transputer operating system, which provides a UNIX-like interface together with a transputer C compiler and library routines to facilitate message passing between processes running on the transputer array.

Various levels of message passing are possible, which trade flexibility and ease of use against message passing speed. The most flexible *transport* procedure allows transparent message passing, ie. message passing which may involve several processor transfers, at a very significant time penalty. The partitioned simulations discussed below were initially undertaken using these functions, but caused the overall simulation to execute *slower* than the single processor simulation! Consequently, nearest neighbour communications between processors, using very low level data transfer functions were used subsequently.

Partitioning

The circuit of Figure 1 was mapped onto two, three, four, six and seven processors. Nearest neighbour data transfer using low level communication was essential to achieve an overall simulation speed up. The circuit and processor topologies therefore limited the possible configurations (the mapping of components to processors).

The actual number of transmission line connections between the partitions was not very significant, as all data transferred between processors was combined into a single message, sent at each time step. In this case, the latency of administering the message itself was far greater than the length of each message. Figures 6 and 7 illustrate examples of the circuit-to-processor mapping which has been used to partition the circuit simulation. Alternative partitioning schemes, also involving two, three, four and seven processors were examined; examples of two processor and four processor configurations are shown in Figures 8 and 9. Figures 6 and 7 are subsequently referred to as the *primary* processor mappings and Figures 8 and 9 as the *secondary* processor mappings.

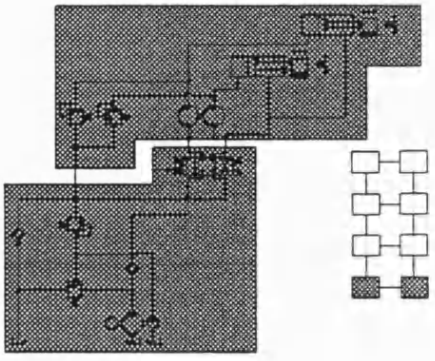


FIGURE 6 2 PROCESSOR PARTITION

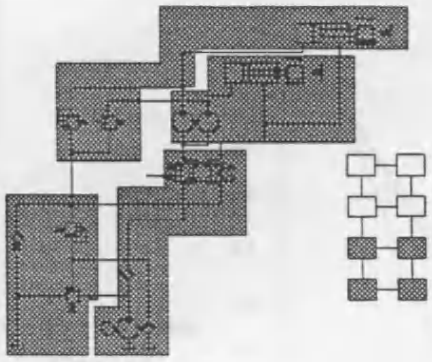


FIGURE 7 4 PROCESSOR PARTITION

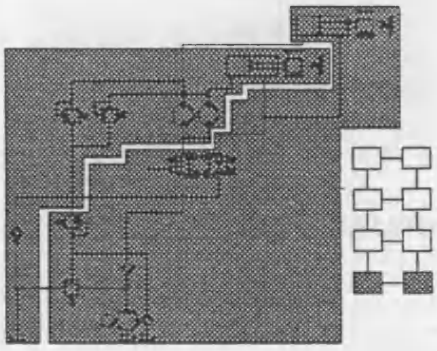


FIGURE 8 SECONDARY 2 PROCESSOR PARTITION

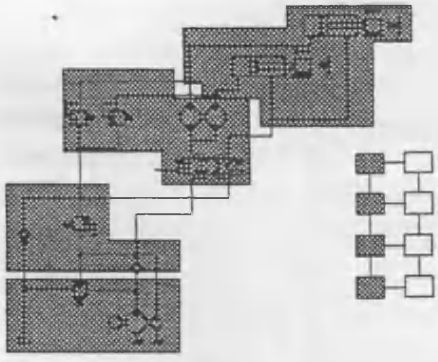


FIGURE 9 SECONDARY 4 PROCESSOR PARTITION

The component models assigned to each partition were divided as evenly as possible, subject to the nearest neighbour constraint, in an attempt to balance the computational load on each processor. This strategy assumes that each model requires roughly the same amount of execution time throughout the simulation.

DISCUSSION

Figure 10 shows the variation of run time reduction (normalised against single processor performance) with the number of circuit partitions (processors) for both the primary, and secondary, model-to-processor mappings. The ideal run time reduction (inverse of number of processors) does not include the inefficiencies introduced when transferring data. The ratio of observed speed up to the ideal speed up (number of processors) is referred to as the simulation *efficiency* and is given in Figure 11. The most efficient *distributed* simulation is the two processor case, at nearly seventy percent, as only one data link is in use and the partitions are comparatively large. However, the less efficient four processor simulation was much faster than the most efficient two processor case (Figure 8), as the increased parallelization was far more significant than the reduced efficiency. For the six processor case (about three components per processor) the drop in efficiency outweighed the increase in distributed processing, the consequence of which was an overall "speed down". The increase in parallel operation to seven processors was almost exactly matched by the further reduction in efficiency, owing to increased communications and reduced computation within each partition.

The secondary partitioning arrangements resulted in slightly improved performance, because of better load balancing of the processors. For example, the primary four partition scheme, consisted of 5,5,6 and 7 component models in each, whereas the alternative partitioning scheme had 6,6,6 and 7 component models on each processor. This adds weight to the initial assumption that each model is roughly equivalent in terms of computational load.

It is interesting to examine the change in performance with data transfer interval. Only the two processor case will be considered here. The time step of 10μs was maintained in each partition, but the data transfer (or data exchange interval) between the processors was reduced successively. Figure 12 clearly indicates the increase in performance possible with reduced communications; the potential problem is of course reduced accuracy. Loss of accuracy was assessed by examining predictions of the inlet and pilot pressure transients of the pilot operated valve, which are quite sensitive to changes in simulation parameters. Noticeable deviations in the transients were observed when the communication interval was increased above 100μs.

Another, more attractive, scheme is to use different time steps in different partitions. For example, a much larger time step is

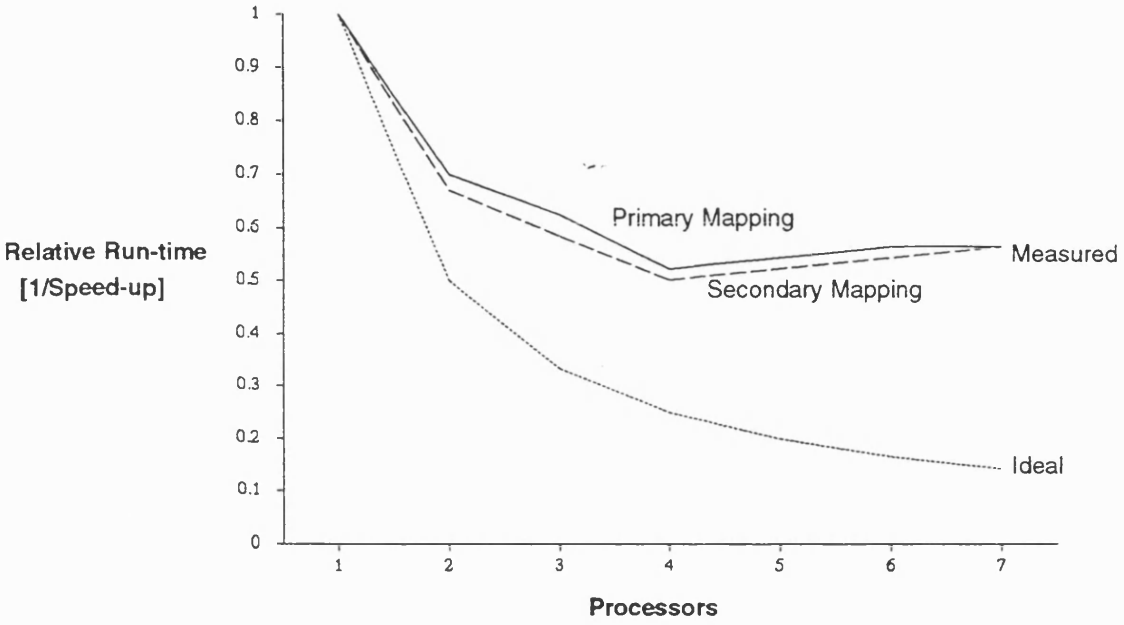


FIGURE 10 RELATIVE RUN-TIME AS A FUNCTION OF PROCESSOR NUMBER

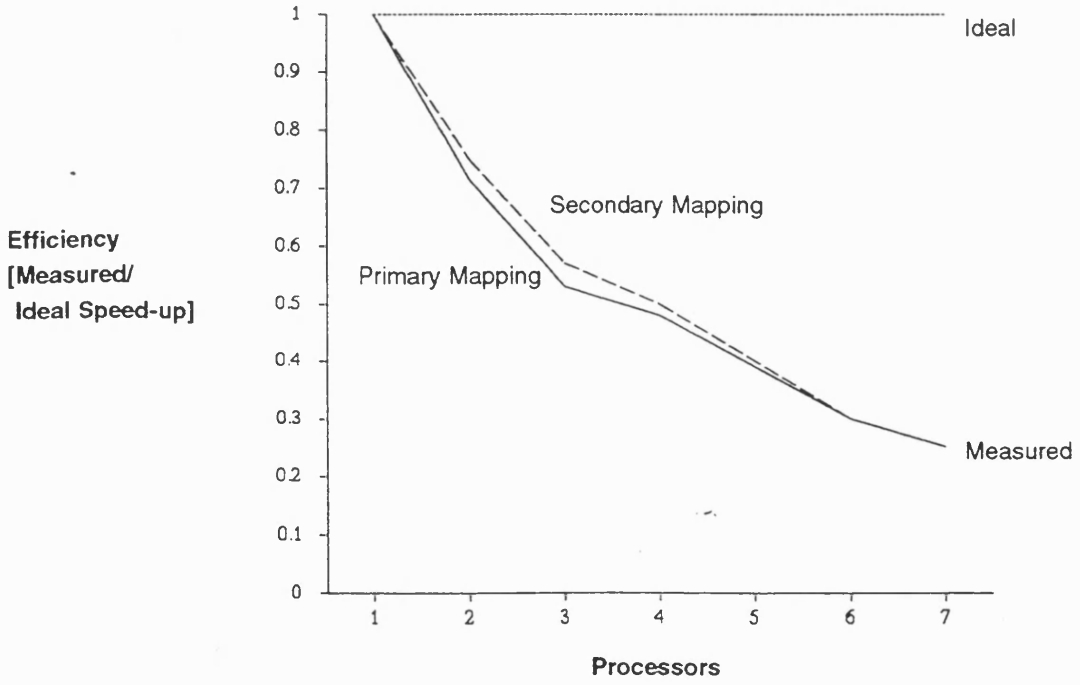


FIGURE 11 PARALLEL EFFICIENCY AS A FUNCTION OF PROCESSOR NUMBER

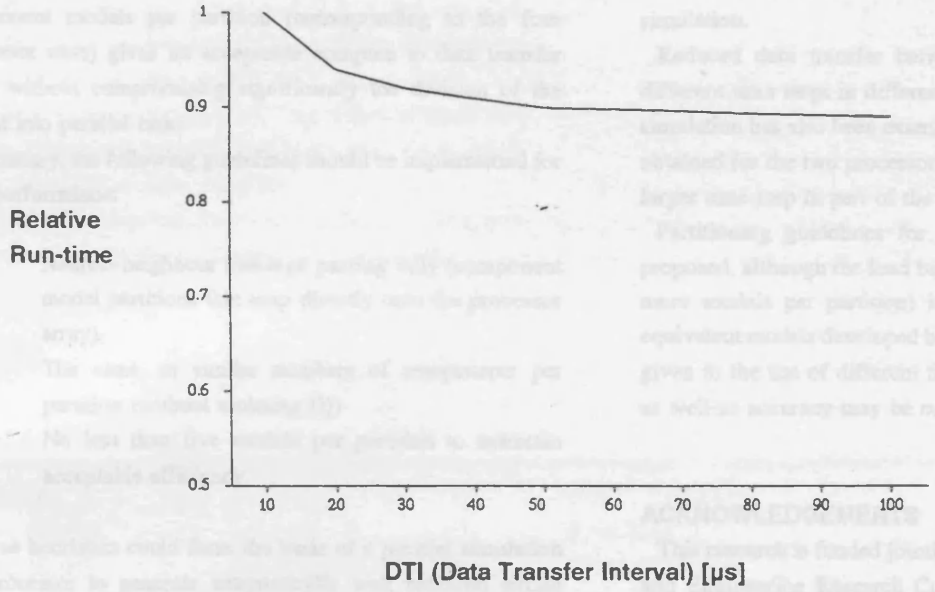


FIGURE 12 2-PROCESSOR RUN-TIME AS A FUNCTION OF DATA TRANSFER INTERVAL (DTI)

tolerated in the actuator partition, because of its relatively slow transient behaviour. The characteristic pressures are propagated along the transmission lines at intervals corresponding to the larger time step in the actuator partition. Between communication time steps in the pump partition, the characteristic pressure signals from the actuator partition are held constant. Owing to the different time steps, the impedances used when transferring characteristic data must be adjusted to compensate. Line impedance is defined as:

$$Z = \frac{\rho c}{A} = \frac{\Delta}{V\!/B_e} \qquad (15)$$

Here two time steps are used, Δ_{p0} in the pump partition (p0) and Δ_{p1} in the other (p1). To propagate characteristics from partition p0 to partition p1:

$$\bar{C}_{p0} = \bar{P}_{p0} + n\bar{Z}_{p0}\bar{Q}_{p0} \qquad (16)$$

and from p1 to p0:

$$\bar{C}_{p1} = \bar{P}_{p1} + \frac{\bar{Z}_{p1}}{n}\bar{Q}_{p1} \qquad (17)$$

Where $n=\Delta_{p0}/\Delta_{p1}$

This technique works very well and dramatically increased the speed up factor to 2.1 for two processors. [Note: the speed up factor is relative to the single processor simulation using 10µs throughout]. An examination of the predicted pilot and inlet pressures to the pilot operated valve, using a 10µs and a 100µs steps in the pump (p0) and actuator (p1) partitions respectively, showed only a very small loss of accuracy.

PARTITIONING GUIDELINES

The speed at which data is transferred between partitions, the relative distribution of computational loads and the proportion of compute time to data transfer time are all important factors in the partitioning process.

As stated earlier, message passing that requires transfers between processors demands, in the system used, a high level of resources, which negates any advantage in parallel computing. The latency of nearest neighbour message passing, however, is very much less.

Computational load balancing of each processor has been attempted here by placing equivalent numbers of component models into each partition. This is only valid because, in the example considered, each model executes almost the same amount of code per step. This division of models must be accomplished whilst retaining the nearest neighbour configuration, ie. that the partitions map directly onto the processor array.

The results of this investigation suggest that a minimum of five component models per partition (corresponding to the four processor case) gives an acceptable compute to data transfer ratio, without compromising significantly the division of the circuit into parallel tasks.

In summary, the following guidelines should be implemented for best performance:

(i) Nearest neighbour message passing only (component model partitions that map directly onto the processor array).

(ii) The same, or similar numbers of components per partition (without violating (i))

(iii) No less than five models per partition to maintain acceptable efficiency.

These heuristics could form the basis of a parallel simulation pre-processor to generate automatically well balanced circuit partitions. One of the simplifications made in assigning models to processing elements was the equivalence, in terms of execution speed, of each model. This assumption is not true of all possible models and more sophisticated processor load estimates for each component model may be needed. Components that use iterative calculation, for example, may require more compute time than components requiring readily computed algebraic solutions.

CONCLUSIONS

A hydraulic circuit simulation using transmission line models of the components has been evaluated and, for a sample circuit, compared with its lumped parameter equivalent with excellent results. [The TLM simulation required approximately one tenth of the CPU for comparable accuracy].

Parallel processor implementation of the procedure has been found to be straightforward, as each component is numerically decoupled from connected components, owing to the propagation delay modelled by the lines. This allows circuit partitions (groups of connected components) to map onto a processor array. The TLM simulation was partitioned onto two, three, four, six and seven processors and a total of nine different partitioning arrangements investigated. The speed up and efficiency of each distributed simulation was quantified for each simulation. Only nearest neighbour communications between processors was permitted, allowing (fast) low level messages to be used. This limited the number of possible configurations, as the circuit partitions had to map directly onto a 2x4 transputer array. The best speed up obtained was almost 2 for the four processor simulation.

Reduced data transfer between processors and the use of different time steps in different partitions for the two processor simulation has also been examined. A speed up factor of 2.1 was obtained for the two processor case, owing to the use of a much larger time step in part of the simulation.

Partitioning guidelines for maximum speed up have been proposed, although the load balancing recommendation (five, or more models per partition) is limited to the computationally equivalent models developed here. Careful consideration must be given to the use of different time steps, as the processor loads, as well as accuracy may be compromised.

ACKNOWLEDGEMENTS

This research is funded jointly by the UK Government Science and Engineering Research Council, SERC Grant GR/F62971, and the Ministry of Defence. The authors gratefully acknowledge this support.

REFERENCES

Boucher R F & Kitsios E E, 1986, "Simulation of Fluid Network Dynamics by Transmission Line Modelling", *Proc. IMechE*, Vol 200, No C1., pp21-29.

Burton J D, Edge K A & Burrows C R, 1992, "Modelling Requirements for the Parallel Simulation of Hydraulic Systems", *ASME WAM*, Anaheim, CA.

Burton J D, Edge K A & Burrows C R, 1993, "Analysis of an Electro-Hydraulic Position Control Servo-System using Transmission Line Modelling", *2nd JHPS International Symposium on Fluid Power*, Tokyo, Japan.

D'azzo JJ & Houpis C H, 1988, "*Linear Control System Analysis and Design*", 3rd Ed., Mc Graw-Hill Int., pp757-762.

D'souza A F and Oldenburger R, 1964, "Dynamic Response of Fluid Lines", *Trans. ASME J. of Basic Eng.*, pp589.

Johns P B & O'Brien M, 1980, "Use of the Transmission Line Modelling (TLM) method to Solve Non-linear Lumped Networks", *The Radio and Electronic Engineer*, Vol 50, No 1/2, pp59-70.

Krus P, Jansson A, Palmberg J-O & Weddfelt K, 1990, "Distributed Simulation of Hydromechanical Systems", 3rd Bath International Fluid Power Workshop. *Research Studies Press*, UK.

McCandlish D & Dorey R E, 1983, "The Mathematical Modelling of Hydraulic Pumps and Motors", *Proc. IMechE*, Vol

198B, N°10, pp165-174.
Richards C W, Tilley D G, Tomlinson S P & Burrows C R, 1990, "Type Insensitive Integration Codes for the Simulation of Fluid Power Systems", *Proc. ASME WAM*, Texas, USA.
Sidell R S & Wormley D N, 1977, "An Efficient Simulation Method for Distributed-Lumped Fluid Networks", *Trans. ASME J. of Dyn. Sys. Mea. & Con.*, pp34-40
Viersma T J, 1980, "*Analysis, Synthesis and Design of Hydraulic Servo-systems and Pipelines*", Elsevier spc, pp145-148.

APPENDIX

act1/2:	**fdiv:**	**cv1:**	**cv2:**
F_S 100 [N]	**m1:**	P_c 10 [bar]	P_c .5 [bar]
F_R 50 [N]	D_m 100 [cc/rev]	K 100 [l/min/bar]	K 100 [l/min/bar]
F_L 100 [N]	c_f 0.1		
d_p 125 [mm]	c_s $3x10^{-8}$	**lines:**	**dcv:**
d_r 70 [mm]	c_v $2x10^5$	d 25.4 [mm]	ΔP_{ref} 205 [bar]
M_e 10 [Kg]	**m2:**	$l_1+l_3+l_7$ 4 [m]	Q_{ref} 5 [l/min]
c 3000 [N/(m/s)]	D_m 100 [cc/rev]	$l_2+l_4+l_5+l_6+l_8+l_9$ 4 [m]	
x_o 1600 [mm]	c_f 0.2	$l_{10}+l_{11}+l_{12}$ 3 [m]	**tk:**
v_o 0 [m/s]	c_s $5x10^{-8}$	$l_{13}+l_{14}+l_{15}+l_{16}+l_{17}+l_{18}$ 4 [m]	P_t 0 [bar]
x_K 1700 [mm]	c_v $3x10^5$	$l_{19}+l_{25}+l_{28}$ 8 [m]	
K 1.8 [KN/mm]		$l_{20}+l_{26}+l_{27}$ 8 [m]	
s 1850 [mm]	**shaft:**	$l_{21}+l_{23}+l_{24}$ 1 [m]	
	J 0.1 [Kg m^2]	$l_{22}+l_{29}+l_{30}$ 12 [m]	
rv1/2/3:	c 0.1 [Nm/rpm]		
P_c 210 [bar]			
K 600 [(l/min)/bar]	**p1:**	**system:**	
	Q_{p1} 121 [l/min]	B_e 8900 [bar]	
prv:		ρ 890 [Kg/m^3]	
P_c 145 [bar]	**p2:**	ν $60x10^{-6}$ [m^2/s]	
K 600 [l/min/bar]	Q_{p2} 84 [l/min]	P_o 0 [bar]	

TABLE 1 SYSTEM PARAMETERS

2/12/92 JMHS version

ANALYSIS OF AN ELECTRO-HYDRAULIC POSITION CONTROL SERVO-SYSTEM USING TRANSMISSION-LINE MODELLING (TLM)

JAMES D BURTON KEVIN A EDGE CLIFFORD R BURROWS
Fluid Power Centre
School of Mechanical Engineering
University of Bath
Bath, UK

ABSTRACT

This paper assesses the potential of a transmission-line modelling approach to the dynamic simulation of electro-hydraulic systems. Numerical models are developed for the case of an electro-hydraulic position control system, which forms the basis for a comparison between lumped parameter integration techniques and distributed parameter wave equation modelling using TLM. The selection of simulation time-step for the TLM case and the effect on results is discussed. Potential difficulties simulating both analogue and digital controllers are investigated and recommendations made for selecting appropriate simulation parameters. In addition, the effects on simulated behaviour of different line models, ie. compressibility-only and distributed friction models, is also examined.

KEYWORDS Transmission-Line Modelling, Electro-Hydraulic Systems, Simulation

NOTATION

A	Pipe cross-sectional area	x	Position
B	Fluid Bulk Modulus	Z	Line impedance
C	Characteristic Pressure	Z_o	Line surge impedance, $\rho c/A$
c	Acoustic velocity $\sqrt{B_e/\rho}$		
	Pump/Motor loss coefficient	Δ	Fixed time step for TLM simulation
	Viscous friction	θ	Angular position
D	Pump/Motor Displacement	μ	Fluid dynamic viscosity
d	Demand signal	ρ	Fluid density
E	Controller error signal	τ	Time constant
f	Function of, Feedback signal	ω	Angular speed
J	Inertia		
K	Valve flow-pressure gradient, Gain	Subscripts	
N	Fluid line loss function	a	Transmission-line end a
P	Pressure	b	Transmission-line end b, Boost
Q	Flow	c	Cracking
r	Transmission reduction ratio	e	Effective
s	Laplace operator	f	Coulomb friction
T	Wave propagation time, Torque	i	Integral, Integer

in	Input
J	Junction
L	Load
m	Motor
out	Output
p	Pump, Proportional
pm	Prime mover
s	Shaft, Servo, Slip
v	Valve, Viscous, Velocity
θ	Positional

Superscripts

in	Input
out	Output

INTRODUCTION

In this paper a technique is presented for the simulation of hydraulic control system performance incorporating the transmission-line modelling (TLM) approach for hydraulic lines [1][2]. An electro-hydraulic position control system outlined in Figure 1 is used as an example to evaluate and compare the method with established lumped parameter numerical integration techniques and to examine possible problems in modelling analogue and digital controllers. The motivation for this work is to demonstrate the effectiveness of the TLM method in solving fluid power problems, by encapsulating all relevant calculations within a particular component model, *including numerical integration*. Such an approach avoids the need for a highly complex, centralized integrator capable of solving numerically stiff differential equations [3] and, although not specifically considered here, such an approach is inherently suited to parallel processing [4]; each component in the system is numerically isolated by at least one propagation delay, equal to the integration time step, allowing the sub-systems to become decoupled.

Components such as pipe junctions, pressure or flow sources, restrictors, valves, actuators etc. are governed by combinations of algebraic, discontinuous and differential equations. Bi-linear transformation (equivalent to trapezoidal integration) is used to solve the component differential equations.

To obtain an explicit solution at each time step, equations relating pressure and flow at each component port are solved simultaneously with the transmission-line wave equations that represent the fluid lines connecting component models in the system. The 4-pole, transmission-line equations are given by eq.(1), where N(s) is a loss function dependent upon the type of friction model employed [5].

$$\begin{aligned} P_a - Z_o\sqrt{N(s)}Q_a &= \left(P_b + Z_o\sqrt{N(s)}\,Q_b\right) e^{-Ts\sqrt{N(s)}} \\ P_b - Z_o\sqrt{N(s)}Q_b &= \left(P_a + Z_o\sqrt{N(s)}\,Q_a\right) e^{-Ts\sqrt{N(s)}} \end{aligned} \quad (1)$$

In the time domain it is generally possible to develop two *transmission-line end* equations of the form:

$$\begin{aligned} P_a - Z_oQ_a &= C_b(t-T) \\ P_b - Z_oQ_a &= C_a(t-T) \end{aligned} \quad (2)$$

Where T is the wave propagation time, L/c. Pressure and flow at one end of the line is related to the past time history of pressure and flow at the opposite line end; this past time-history is referred to as the characteristic pressure, denoted C.

HYDRAULIC TRANSMISSION-LINE MODELS

For a lossless pipeline N(s) in eq.(1) equals unity; the characteristic pressures are then simply:

$$\begin{aligned} C_a(t-T) &= P_a(t-T) + Z_oQ_a(t-T) \\ C_b(t-T) &= P_b(t-T) + Z_oQ_b(t-T) \end{aligned} \quad (3)$$

Here the propagation operator is only a transmission delay of

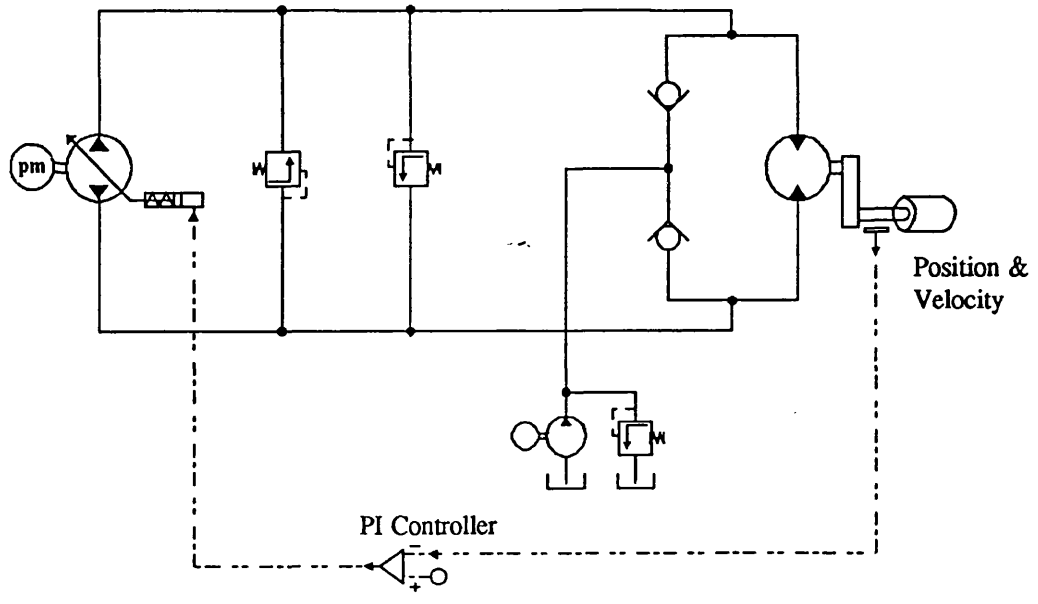


Figure 1 Electro-Hydraulic Position Control System

duration T, which decouples the component models that form the numerical simulation.

Compressible Pipeline

For the lossless case, an approximation to a compressible pipeline is obtained by setting the transmission delay, T, to the global integration time-step, Δ. This implies a modified line length and consequently the pipe cross-sectional area must be adjusted to compensate, thereby ensuring correct modelling of compressibility effects, as the capacitive term (V/B_e) remains constant. The time-step Δ must be relatively small, otherwise the line dynamics will become excessively distorted due to the inertance term ($\rho L/A$) becoming dominant. However, some fluid inertance is always present and is a realistic effect.
If unrealistic resonant effects are to be avoided, low pass filtering of the characteristics is useful and, in fact, approximates the frequency-dependent damping evident in actual pipelines. The filtering method described by Krus et al [6] has been used in all of the simulations presented in this paper.

Long Lines

In this case the line may be effectively discretized into an integer number of integration time steps, retaining the actual line delay, T. Various line models are possible, ranging from distributed friction approximations [7] to frequency-dependant friction algorithms incorporating convolution integrals of the complete characteristic time history [2]. The distributed friction line model proposed in [7] is used in this paper.

THE APPROACH TO SYSTEM SIMULATION

A complete system simulation is facilitated by linking the various component models together within a control program, which inter alia calls the models in a pre-defined sequence. Component simulation data required for plotting are also manipulated and stored for later inspection.
Information transfer between connected models is handled differently, depending upon the type of component port, in this case either *hydraulic* or *signal* ports. Signals are uni-directional and are therefore propagated from one model to the next *instantaneously*, achieved by ensuring that the correct call sequence is observed. This sequence is:

1. **Controller** input demand duty cycle (signal input to controller)
2. **Motor-gearbox-load** model (velocity & position transducer signal)
3. **Sample & Hold** (digital controller only)
4. **PI Controller** (demand and feedback signals in and swash servo signal output)
5. **Variable Displacement Pump** (swash servo input)
6. **Other components** (arbitrary order) ...

Inputs to hydraulic ports are delayed characteristic pressures and are thus unaffected by the call sequence, except, of course, if the component also has electrical signal links. Initial values for the characteristic pressure at each hydraulic port must be specified to initiate the calculation process and this is normally achieved by setting a constant pressure starting condition (zero initial flows). In addition, due to the unique flow symmetry of the transmission line element, hydraulic component ports may be linked consistently, as the characteristic pressures which couple the hydraulic ports are scalar quantities.
Although not directly relevant to this investigation it is important to be able to model cavitation; a suitable approach based on column separation is described in [4].

COMPONENT MODELS

Hydraulic Components

For each component connected between transmission-lines, the following *system* of equations must be solved. [Note: the modelling convention used is such that each component port is connected to a transmission line at end a]

The component equation:

$$\bar{Q}_a = f(\bar{P}_a) \tag{4}$$

[where the *bar* denotes a vector quantity]

and the transmission-line end equations at each component port:

$$\bar{P}_a = \bar{Z}_o \bar{Q}_a = \bar{C}_b e^{-s\bar{T}} \tag{5}$$

The component equation is solved simultaneously with the transmission-line end equations at each port of the component, whilst ensuring that the flow convention is observed. The symmetrical configuration (positive flows into each end of the line) is essential, as the component ports of models in a system may then be linked consistently.
The following hydraulic components are used in the simulation of the example electro-hydraulic circuit, shown in Figure 1.

Junction Model. To assemble circuits of the complexity required in most real applications, junction (node) models are needed. Taken in conjunction with the compressible line approximation, the junction model represents an equivalent fluid volume with multiple (in this case three) ports. Figure 2 is an example of a 3 port node.

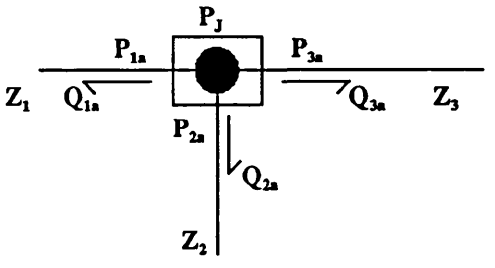


Figure 2 Hydraulic Junction

Junction equations:

$$P_J - P_{1a} - P_{2a} - P_{3a} \qquad \sum_{i=1}^{3} Q_{ia} = 0 \tag{6}$$

Transmission-Line equations (i=1, 2, 3):

$$P_{ia} - Z_i Q_{ia} = C_{ib} e^{-sTi} \tag{7}$$

Solving simultaneously to obtain the junction pressure:

$$P_J = Z_J \sum_{i=1}^{3} \frac{C_{ib} e^{-sTi}}{Z_i} \qquad \frac{1}{Z_J} = \sum_{i=1}^{3} \frac{1}{Z_i} \tag{8}$$

and subsequently for the flow at each port (i=1, 2, 3):

$$Q_{ia} = \frac{1}{Z_i}\left(P_J - C_{ib}e^{-sT_i}\right) \qquad (9)$$

The characteristic pressures $C_{ia}=P_J+Z_iQ_{ia}$ at each port are then propagated as inputs to connecting models for the next time-step, t+Δ. These equations are easily extended to incorporate any number of ports.

Relief/Check Valve Model. The simplest model of a relief valve, or check valve (Figure 3) assumes that the valve dynamics are instantaneous. For this type of model the valve is uni-directional and no flow across the valve takes place until the differential pressure exceeds the cracking pressure, P_c. A linear flow-pressure characteristic of gradient K is assumed once the valve has cracked open.

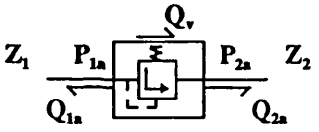


Figure 3 Relief Valve Model

Transmission-line end equations at each valve port:

$$\begin{aligned} P_{1a} &= C_{1b}e^{-sT1} - Z_1Q_v \\ P_{2a} &= C_{2b}e^{-sT2} + Z_2Q_v \end{aligned} \qquad (10)$$

Equations relating valve flow to differential pressure are:

$$\begin{aligned} (P_{1a}-P_{2a}) > P_c \qquad & Q_v = K(P_{1a}-P_{2a}-P_c) \\ (P_{1a}-P_{2a}) \le P_c \qquad & Q_v = 0 \end{aligned} \qquad (11)$$

Flow is determined explicitly by substitution of the pipe-end (characteristic pressure) equations into the valve equation, as follows:

$$\begin{aligned} P_{1a}-P_{2a} > P_c \qquad & Q_v = \frac{K\left(C_{1b}e^{-sT1} - C_{2b}e^{-sT2} - P_c\right)}{1+K(Z_1+Z_2)} \\ P_{1a}-P_{2a} \le P_c \qquad & Q_v = 0 \end{aligned} \qquad (12)$$

Valve port pressures and characteristic pressures are then readily determined from Q_v.

Motor-Gearbox-Load Model. This is a *lumped-distributed* model of a hydraulic motor, with leakage and friction losses [8], coupled via a rigid shaft to a reduction gearbox and an inertial plus viscous load (Figure 4). The *distributed* part of the description refers to the hydraulic ports (transmission-line ends) connected to a *lumped* model of the motor mechanism, gear reduction and load. To obtain a discrete solution the lumped component must be solved simultaneously with the line ends. The differential equation describing the load is converted to an implicit difference equation using bi-linear transformation, equivalent to trapezoidal integration; the resulting equations are solved algebraically, as before.

Motor ideal flow plus leakage flow:

$$Q_m = D_m\omega_m + \frac{c_sD_m}{\mu}(P_{1a}-P_{2a}) \qquad (13)$$

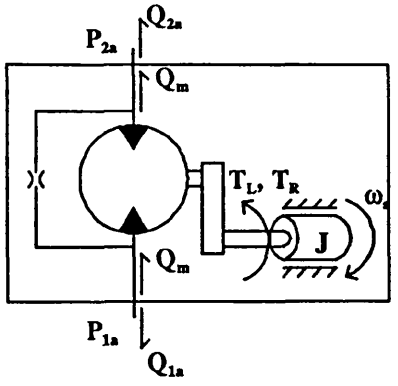


Figure 4 Hydraulic Motor-Gearbox-Load Model

Motor Torque, including coulomb and viscous friction losses:

$$T_m = D_m(1-c_f)(P_{1a}-P_{2a}) - c_vD_m\mu\omega_m \qquad (14)$$

Idealized transmission (lossless, infinitely stiff & zero backlash):

$$T_s = T_mr \qquad \omega_s = \frac{\omega_m}{r} \qquad (15)$$

Line-end equations at each port:

$$\begin{aligned} P_{1a} &= C_{1b}e^{-sT1} - Z_1Q_p \\ P_{2a} &= C_{2b}e^{-sT2} + Z_2Q_p \end{aligned} \qquad (16)$$

Mechanical load equation:

$$Js\omega_s = T_S - T_L - T_R sgn(\omega_s) - c_L\omega_s \qquad (17)$$

First, derive an expression for the motor differential pressure:

$$(P_{1a}-P_{2a}) = \frac{(C_{1b}-C_{2b}) - rD_m\omega_s(Z_1+Z_2)}{1 + (Z_1+Z_2)\dfrac{c_sD_m}{\mu}} \qquad (18)$$

Further algebraic manipulation of the above equations results in an expression for output shaft speed:

$$\begin{aligned} Js\,\omega_s = & \frac{rD_m(1-c_f)}{1 + (Z_1+Z_2)\dfrac{c_sD_m}{\mu}}\left(C_{1b}e^{-sT1} - C_{2b}e^{-sT2}\right) - \\ & \left(\frac{r^2D_m^2(1-c_f)(Z_1+Z_2)}{1 + (Z_1+Z_2)\dfrac{c_sD_m}{\mu}} + c_vD_m\mu r^2 + c_L\right)\omega_s - \\ & T_L - T_R sgn(\omega_s) \end{aligned} \qquad (19)$$

and for the position:

$$\theta_s = \frac{\omega_s}{s} \qquad (20)$$

Subsequently, it is quite straightforward to apply the bi-linear transform and obtain a form suitable for digital computation of ω_s and θ_s. Back substitution gives $(P_{1a}-P_{2a})$, Q_m, T_s, T_m and finally P_{1a} and P_{2a}. The characteristic pressures, $C_{1a} = P_{1a}-Z_1Q_m$ and $C_{2a} = P_{2a}+Z_2Q_m$, are then calculated for propagation to connecting component ports at the next time step.

Controllers and Electro-Hydraulic Components

For the circuit considered in this paper, components such as the motor-load (incorporating transducer signals), the controller (including the demand duty cycle) and the swash-controlled variable displacement pump all involve the transmission of electrical signals. The motor-load model calculates the output shaft acceleration, velocity and position. The velocity and position are transmitted as *instantaneous* signals to the controller model, which in turn operates on these signals to provide an *instantaneous* output signal to the pump swash controller. Signal propagation is achieved using the correct component model call sequencing within each time step. Conversely, hydraulic port connections are provided with output characteristic pressures from the ports of connected models, calculated at the previous time step.

Proportional plus Integral (PI) Controller. The controller shown in Figure 5 has an input demand signal **d** and a feedback signal **f**. For the example system **f**, is the sum of the position and velocity feedback signals, provided by the motor-load model. E_c refers to the compensated error, propagated *instantaneously* to the pump swash controller.

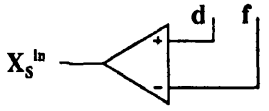


Figure 5 PI Controller

Feedback signal:

$$f = K_\theta \theta + K_v v \tag{21}$$

Error:

$$E = K_{in}(d-f) \tag{22}$$

PI controller transfer function:

$$X_s^{in} = K_{out}\left(K_p + \frac{K_i}{s}\right)E \tag{23}$$

Sample & Hold. For the case of the digital PI controller model a sampling mechanism is required. This is achieved numerically by *holding* the control signal for **n** fixed time steps of duration, Δ, resulting in a sampling interval of $n\Delta$.

Variable Displacement Pump. This model (shown in Figure 6) assumes that a constant speed prime mover drives a pump incorporating friction and leakage losses [8]. The pump model is bi-directional and incorporates the dynamics of a swash servo, represented by a first order lag. Pump inlet and outlet are modelled as transmission-line ends. The input to the swash servo is provided *instantaneously* by the controller output, at each time step.

Pump swash servo mechanism:

$$X_S^{out} = \frac{X_S^{in}}{\tau_s s + 1} \tag{24}$$

The pump flow equation, including losses:

$$Q_p = X_s^{out} D_p \omega_{pm} - \frac{c_s D_p}{\mu}(P_{1a} - P_{2a}) \tag{25}$$

Transmission-line end equations at each port:

$$\begin{aligned} P_{1a} &= C_{1a}e^{-sT1} - Z_1 Q_p \\ P_{2a} &= C_{2b}e^{-sT2} + Z_2 Q_p \end{aligned} \tag{26}$$

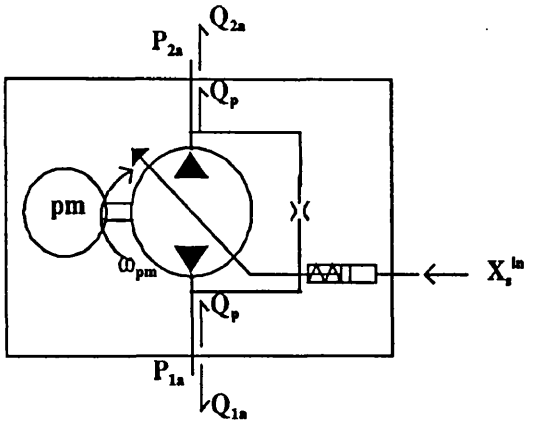


Figure 6 Variable Displacement Pump

Hence, the pump flow, Q_p, in terms of the characteristic variables, C_{1b} and C_{2b}:

$$Q_p = \frac{X_S^{out} D_p \omega_{pm} - \frac{c_s D_p}{\mu}\left(C_{1b}e^{-sT1} - C_{2b}e^{-sT2}\right)}{1 - (Z_1 + Z_2)\frac{c_s D_p}{\mu}} \tag{27}$$

After transformation into discrete time, X_S^{out} and subsequently the other model variables, Q_p, P_{1a}, P_{2a}, C_{1a} and C_{2a} may be evaluated at each time step.

SIMULATIONS

The numerical experiments to be considered examine the response of the system (Figure 1) to a step in positional demand from zero to 180°. All state variables, eg. line end pressures, line end flows, displacements, velocities and accelerations were initialized to zero.

The parameters used for the simulation study, given in Appendix 1, serve only as an example to illustrate the different modelling approaches.

DISCUSSION

Lumped Parameter and Distributed Parameter (TLM) Simulation (Analogue PI Control)

Figures 7a to 7c give some of the salient response characteristics of the TLM circuit model, using a small time step of 10μs. Figure 7a shows the smooth positional response of the inertial load to a the step change in demand. Time responses for the fractional pump displacement and the pressure at the hydraulic motor inlet are given in Figures 7b and 7c, respectively. The results are virtually identical to those obtained from a lumped parameter simulation of this system, obtained using the Bath*fp* package [3]. A ten-fold increase in the TLM simulation step size to 100μs had negligible effect on the nature of the predicted behaviour. However, much larger time steps, of the order of 1ms, or greater, result in significant deviations. For these time steps the transmission lines no longer represent compressibility approximations, but include significant inertia effects; they are therefore no longer comparable to the lumped parameter results.

If the line delay is such that the step size is too large, it is

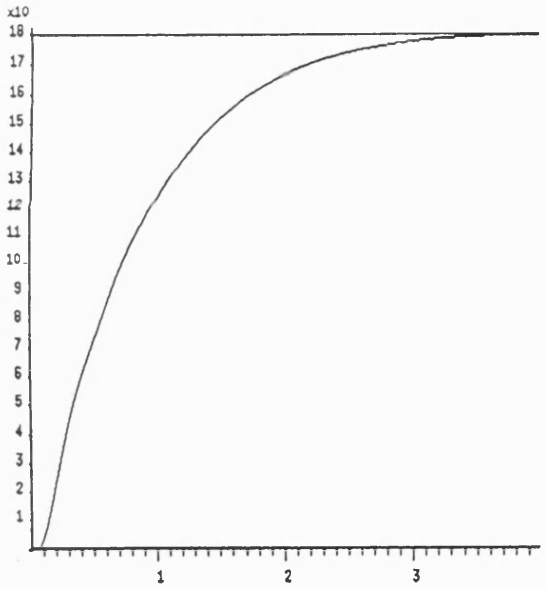


Figure 7a Load Position [deg] vs. Time [s]

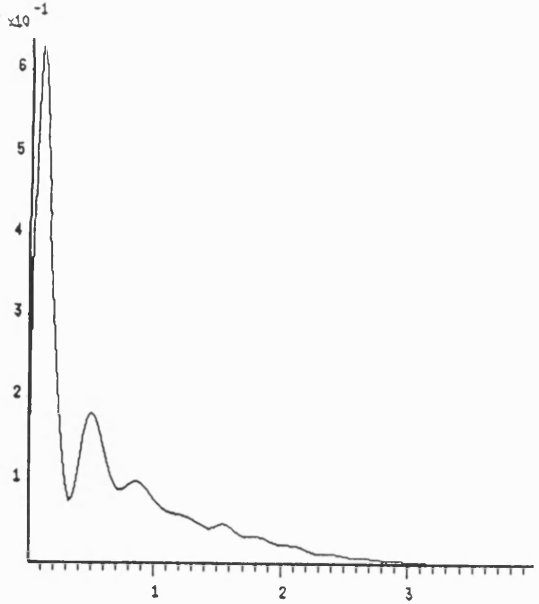


Figure 7b Pump Swash Ratio [] vs. Time [s]

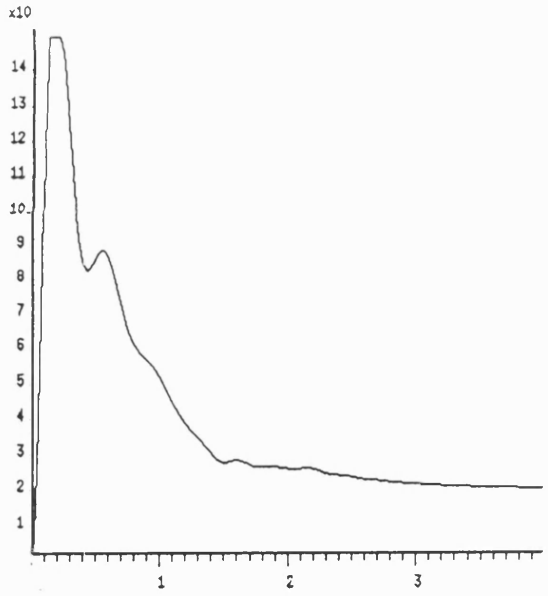


Figure 7c Motor Inlet Pressure [bar] vs. Time [s]

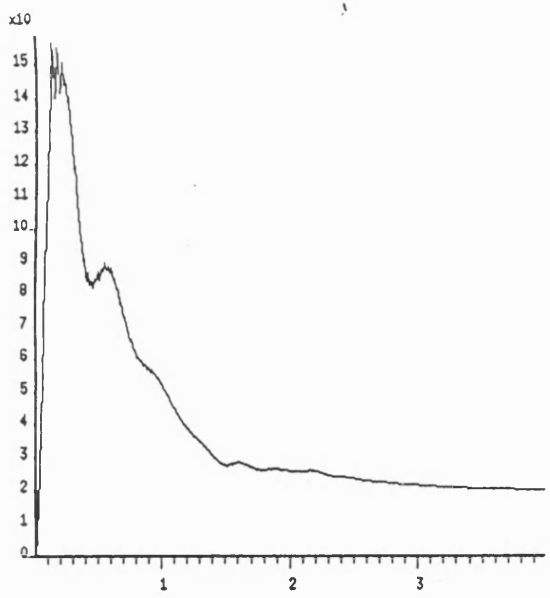


Figure 8 Motor Inlet Pressure [bar] vs. Time [s]: 10m Distributed Friction Lines

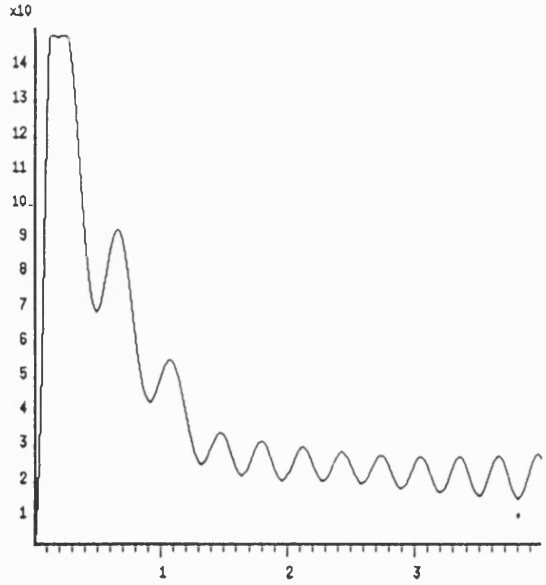


Figure 9 Motor Inlet Pressure [bar] vs. Time [s]: 50 ms Sample Interval

possible that rapid transients occurring at either end of the line will be "sampled" at too low a rate, leading to distortion. To understand the introduction of this distortion due to false inertia effects, it may be helpful to consider the transmission line as the sum of a *lumped* compressibility term (V/B_e) and a *lumped* inertia term ($\rho L/A = \rho c^2\Delta^2/V$). The **additional** pressure due to the fluid inertance is the product of the inertia term and the total rate of change of flow into the line. Thus very rapid flow transients, coupled with a high fluid inertance will result in a correspondingly large and incorrect pressure change.
In the system studied here, the shortest line length was **0.67m**, corresponding to a line delay of **0.67ms**. According to the simulation test results, a time step less than, or equal to about one tenth of the shortest line delay appears to be a convenient and reasonably conservative starting point for step size selection. The fluid inertance term, proportional to the square of the time step, is in this case, likely to be negligibly small and the compressible line approximation is therefore valid. Furthermore the "sampling" interval is then sufficiently small in relation to the modelled line dynamics to avoid distortion. Alternatively, the line may be discretized and modelled as a distributed friction line (with some computational overhead). This ensures a higher "sampling" interval, in addition to more accurate modelling of wave effects in the line.

TLM Distributed Friction Line Model

A **10m** distributed friction transmission line was introduced either side of the hydraulic motor model, whilst all other lines were modelled by capacitive approximations. A simulation time step of **10μs** was used, to avoid distorting excessively the compressible line elements in the system. The additional transmission delay (approx. **10ms**) introduced by the distributed friction line did not adversely affect the predicted load motion, but previously unmodelled wave effects were introduced into the motor inlet pressure response (Figure 8).

Digital PI Controller Model

For the digital controller two digital sampling rates were considered. A simulation time step of **10μs** was used in each simulation. The results for the **10ms** sampling interval were indistinguishable from those given in Figures 7a to 7c. However, with a sampling period of **50ms**, the system was unstable. This instability is clearly exhibited in the motor inlet line pressure response, shown in Figure 9. Little deviation in these results is observed with an order of magnitude increase in time step ($\Delta = 100\mu s$). Further orders of magnitude increases in time step ($>=1ms$), however, invalidate the premise of compressibility-only lines and the results are, therefore, no longer comparable. The dynamics of the line system change significantly with the larger time steps.
When using digital controller models with TLM circuit simulations it is prudent firstly to verify the dynamics of the equivalent analogue circuit model, using a sufficiently small time step in accordance with the above guidelines. The digital sampling interval must then be an integer multiple of this time step.

CONCLUSIONS

1. This paper has demonstrated the suitability of transmission-line simulation methods to hydraulic system modelling, including systems with control elements. Provided that component models are called in the correct sequence, electrical signals may be propagated *instantaneously* between component models.

2. There is a close similarity between the TLM solution and the lumped parameter test case, despite the very significant modelling differences. The transmission line approach, however, has clear advantages in simulating wave propagation effects, as the transmission delay, realistic fluid inertia and distributed friction are incorporated as a consequence of the modelling technique.

3. For fixed time step transmission-line simulation the global simulation time step, **Δ, must** be sufficiently small to match the compressible effects in the line to an acceptable level of accuracy. From the simulation studies **Δ=T/10** for the shortest line length is a convenient and reasonably conservative heuristic, to ensure adequate "sampling" of system dynamics and to minimise the effect of fluid inertia in the case of the compressible line approximation. This time step may need adjustment to accommodate distributed friction lines and digital sampling models, which must operate over an integer number of time steps.

REFERENCES

1. Boucher R F & Kitsios E E, Simulation of Fluid Network Dynamics by Transmission Line Modelling, Proc. IMechE, Vol 200 N° C1, 1986, pp21-29

2. Sidell R S & Wormley D N, An Efficient Simulation Method for Distributed-Lumped Fluid Networks, Trans. ASME J. of Dyn. Sys. Mea. & Con., 1977, pp34-40

3. Richards C W, Tilley D G, Tomlinson S P & Burrows C R, Type Insensitive Integration Codes for the Simulation of Fluid Power Systems, ASME WAM, Dallas, Texas, USA, 1990

4. Burton J D, Edge K A & Burrows C R, Modelling Requirements for the Parallel Simulation of Hydraulic Systems, ASME WAM, Anaheim, CA, USA, 1992.

5. Viersma T J, Analysis, Synthesis and Design of Hydraulic Servo-systems and Pipelines, Elsevier spc, 1980, pp145-148

6. Krus P, Jansson A, Palmberg J-O & Weddfelt K, Distributed Simulation of Hydro-Mechanical Systems, 3rd Bath International Fluid Power Workshop, Research Studies Press, 1991

7. Krus P, Weddfelt K & Palmberg J-O, Fast Pipeline Models for the Simulation of Hydraulic systems, ASME WAM, Atlanta, Georgia, 1991

8. Mc Candlish D & Dorey R E, The Mathematical

Modelling of Hydraulic Pumps and Motors, Proc. IMechE, vol. 198B, N°10, 1983, pp165-174

APPENDIX 1 SIMULATION PARAMETERS

pump/servo:
D_p = 100 [ml/rev]
τ_s = 75 [ms]
$c_s = 1x10^{-8}$
ω_{pm} = 1500 [rpm]

cross-line relief valves:
P_c = 150 [bar]
K = 600[l/min/bar]

check valves:
P_c = 0.5 [bar]
K= 100 [l/min/bar]

motor-load:
r = 20:1
J = 15 [Kg m^2]
D_m = 50 [ml/rev]
$c_s = 2x10^{-8}$
c_L = 50 [Nm/rpm]
c_v = 8.4e5
c_f = 0.02
T_R = 0 [Nm]
T_L = 0 [Nm]

tank:
P_t = 0 [bar]

lines:
L_s = 4+10 [m]
d_s = 25.4 [mm]
L_r = 4+10 [m]
d_r = 25.4 [mm]
L_b = 3.3 [m]
d_b = 25.4 [mm]

V_s = 7094 [ml]
V_r = 7094 [ml]
V_b = 1687 [ml]
B_e = 8900 [bar]

controller:
K_{in} = 1
K_{out} = 1/180
K_p = 1
K_i = 2
K_θ = 1
K_v = 0.8333

boost flow:
Q_b = 1 [l/min]

boost relief:
P_c = 15 [bar]
K = 600[l/min/bar]

The Third Scandinavian International Conference on Fluid Power

Computational Load Balancing of Parallel Hydraulic Circuit Simulations Employing Variable Time Step Transmission Line Modelling

J D Burton K A Edge C R Burrows

Fluid Power Centre
University of Bath
Bath, UK

ABSTRACT

This paper describes the application of transmission line modelling with variable time steps. Simulation performance has been further enhanced by dividing the circuit simulation into separate sub-systems according to circuit connectivity and executing these partitions on different processors in parallel. A 'master-slave' processor configuration was adopted, consistent with the use of a centralized error and step size controller, ensuring the same results for single and multi-processor simulations.
The efficacy of this approach has been demonstrated using an example hydraulic circuit simulation, using one, two, three and four processor configurations. Efficient parallel operation is shown to be highly dependent upon the relationship between communications (data transfer) and component model processing within a sub-system, in conjunction with an acceptable balance of computational loads between the respective circuit partitions.

NOTATION

A	Pipe cross-sectional area
B	Bulk Modulus
c	Acoustic velocity
C	Capacitance
I	Inertance
L	Line length
N	Fluid line loss function
P	Pressure
Q	Flow
s	Laplace operator
T	Wave propagation time
V	Line volume
Z	Line impedance
ρ	Fluid density

Subscripts

a	Line end 'a'
b	Line end 'b'
e	Effective

INTRODUCTION

Transmission line modelling (**TLM**) of the pipelines in fluid power systems provides a unique mechanism for decoupling component models within a circuit simulation, due to the modelling of the propagation delay in the line [Sidell & Wormley, 1977][Krus et al, 1990][Burton, Edge & Burrows, 1992]. The 4-pole, transmission line equations are shown in eq.(1).

$$P_a - ZQ_a = \left(P_b + Z\sqrt{N(s)}Q_b\right)e^{-Ts\sqrt{N(s)}}$$
$$P_b - ZQ_b = \left(P_a + Z\sqrt{N(s)}Q_a\right)e^{-Ts\sqrt{N(s)}} \quad (1)$$

Figure 1 Symmetrical Transmission Line

where N(s) is a loss function dependent upon the type of friction modelled [Viersma, 1980].
Each transmission line termination is solved simultaneously with component equations that relate pressure and flow at each port of the component model. Completely self-contained models may be developed, which have the potential to be executed in parallel, as model inputs are delayed outputs from connected models. The symmetry of the transmission line equations enables component models to be linked consistently when building a circuit model.
Pipeline networks that connect several component models are handled using a junction (or node) model. The line junction is treated in the same way as any other component model. By summing the flows to zero and solving simultaneously with the transmission line ends connected to each junction port, explicit equations for junction pressure and the corresponding flows are obtained [Burton, Edge & Burrows, 1992]. The characteristic pressures (equal to **P+ZQ** for the *lossless* line) at the node are then transmitted to connecting component ports at the next time step.
For many hydraulic system models a simple capacitive pipeline model is sufficient to capture the salient dynamics. When using **TLM** to describe the pipelines the *lossless* form of the transmission line equations (**N(s)=1**) is used to isolate numerically each component model (pump, valve, actuator etc.). To achieve a capacitive approximation the line lengths are adjusted such that all components in the system are isolated by the same propagation delay, thus ensuring synchronisation between all component models in the system [Krus et al, 1990]. For constant line capacitance ($C=V/B_e$) this is manifested by a corresponding change in line impedance ($Z=\rho L/A=T/C$). In order to reduce the effect of resonant oscillations associated with lightly damped wave propagation the low-pass filtering method of Krus et al [1990] has been employed; this is an approximation to the frequency-dependent friction evident in real fluid transmission lines.
As the simulation proceeds, any source of pressure difference between each end of a lossless transmission line must be due to the presence of the fluid inertance term ($I=\rho L/A$) implicit in the **TLM** solution. Any flow transient introduced into the system therefore results in a related transient pressure difference. Fluid inertance is representative of the physical system, although an excessive simulation time step may exaggerate its effect greatly, because the inertance is proportional to the square of the time step ($I=\rho L/A=\rho c^2T^2/V$).
The pressure *error* may be controlled by the **TLM** solver by changing the time step (line delay) during the simulation, whilst maintaining a constant capacitance, **C**. With the aim of reducing simulation run times, Jansson et al [1992] proposed a variable time step **TLM** solver based on this approach and the previous work of Krus, Weddfelt & Palmberg [1990] and Pulko et al [1990].

There are very significant non-linearities evident in fluid power system models, including discontinuities, such as valves opening/closing and actuator end-stop limits. For example, if an actuator hits its end stop the piston velocity and therefore the flows at each port will undergo an abrupt change. Such discontinuous flow transients may be detected by the step-size controller as a pressure difference. The time step is then reduced, or rejected accordingly to maintain this pressure difference within a predefined maximum limit, as described by Jansson et al [1992]. In the scheme used here a central error controller monitors the pressure *errors* associated with all lines to determine the maximum step size allowable.

PARALLEL SIMULATION

Significant reductions in execution time for large-scale, complex hydraulic circuits are feasible using multiple processors. However, the partitioning problem (how models are assigned to processors) is non-trivial. The number of processors and the nature of the partitioning results in many possible configurations. To obtain the optimum increase in performance the computational loads on each processor must be adequately balanced, whilst minimising the penalty associated with data transfer between processors. The use of a variable time step **TLM** solver in conjunction with distributed processing should enable much greater simulation speeds, by both distributing the component models and using the maximum step size for a given accuracy. In the implementation developed here a *master-slave* configuration is adopted (see Figure 2).

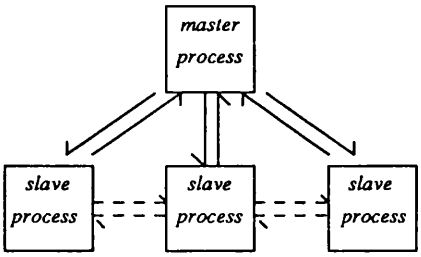


Figure 2 ***Master-Slave*** **Process Configuration**

A single *master* processor is dedicated to error control and step size selection, as well as executing some of the component models, whilst the other *slave* processors execute component models only. The *slave* processors communicate pressure errors to the *master* step size controller, which determines the step size for the entire simulation and therefore ensures synchronisation between processors. Delayed pressure and flow information (RHS eq.(1)) is still exchanged between all of the sub-systems directly, according to the circuit topology. The *master* process only collates the pressure errors in each transmission line and on the basis of the maximum error, determines if the current step is acceptable, or if the step is to be repeated with a smaller time step. In either case the error controller calculates an appropriate step size with which to continue, or repeat the computation, depending upon the relative magnitudes of past and present maximum pressure error [Hairer & Wanner, 1991]. Jansson et al [1992] examined the use of different time steps in two different partitions on the same processor, communicating via shared memory. This approach required two separate step size controllers in each partition. However, severe oscillations in step size selection may result, as the step length controller attempts to synchronise the separate sub-systems at the communication time steps. This technique also requires the step size rejection capability to be deactivated, as a step must be rejected for every subsystem partition for the solution to be consistent numerically. Loss of the ability to reject time steps may impair accuracy, as fast transients and sharp discontinuity points

may not be detected rapidly enough.

The use of a central error controller overcomes these difficulties, although the time step size is then limited to that required by the fastest transient in the entire system. However, process synchronisation is always maintained and the results obtained from a multi-processor simulation will always be identical to those of a single processor. Moreover, as the time step is set throughout the parallel system model the efficiency of the partitioning arrangement chosen is less susceptible to variations in duty cycles in different parts of the parallel simulation. To achieve computational load balancing with a different step sizes on different processors would require the dynamic re-assignment of code during the simulation; with current multi-processing software this is not a very efficient process.

EXAMPLE SYSTEM

In order to demonstrate the efficacy of this approach to parallel simulation, the circuit shown in Figure 3 is used as an example to illustrate the partitioning problems encountered to achieve maximum improvement in simulation speed.

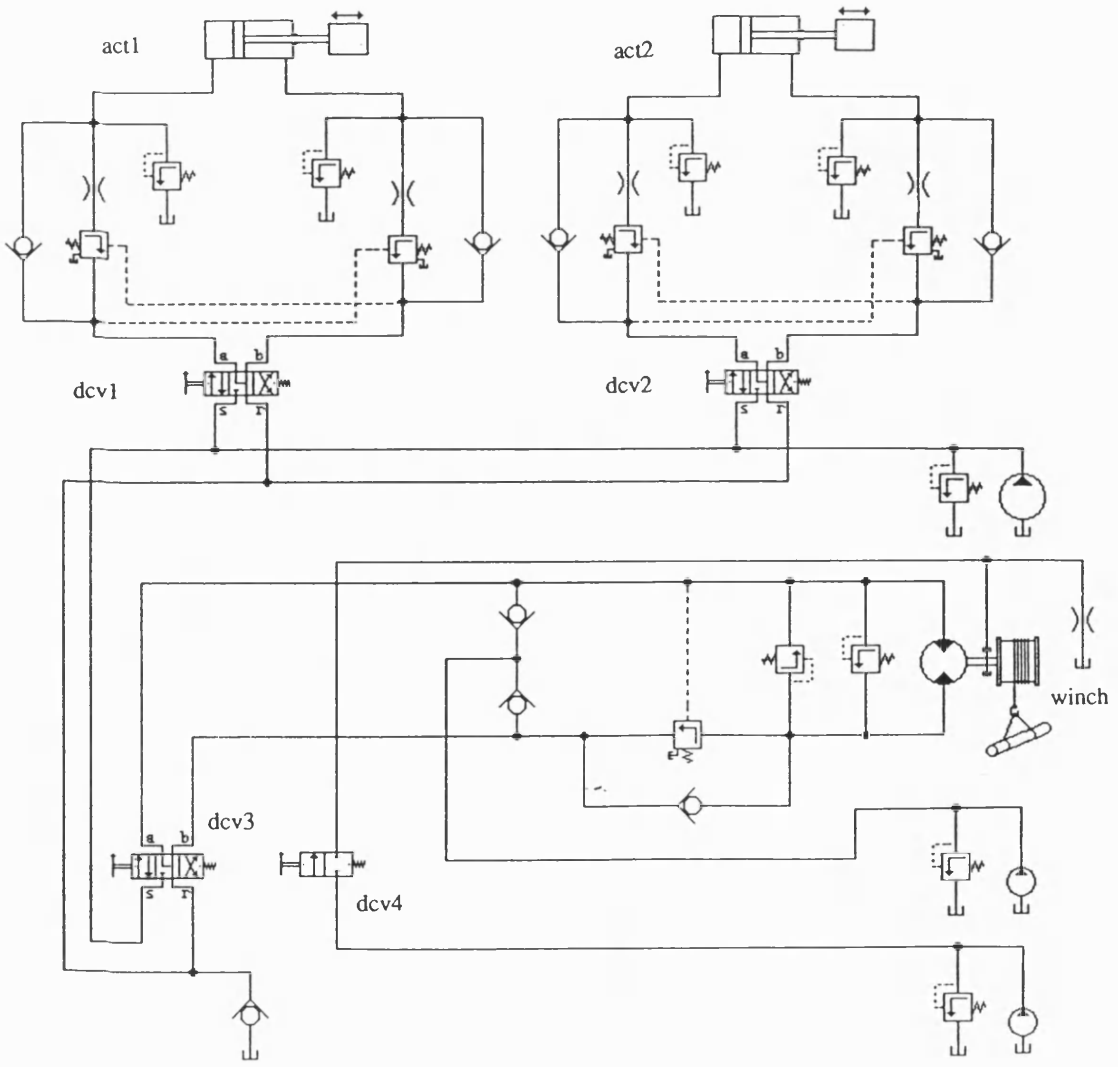


Figure 3 Example System

This circuit is derived from a marine application, involving the operation of two actuators with meter-out flow control and an accompanying closed loop hydrostatic transmission circuit, powered via the vessel's ring main. Most of the component models include significant non-linearities. The actuator model incorporates stiction and coulomb friction, in addition to the travel limits and a time-varying load force. These are also included in the hydraulic motor-load model (travel limits correspond to maximum and minimum cable lengths) combined with a time-varying braking force. Relief and check valve models also exhibit discontinuous behaviour, when opening/closing at a predefined cracking pressure, as well as uni-directional operation. In addition, the directional controls include a square-law orifice model for the flow passages and a time-varying input position.

SINGLE PROCESSOR SIMULATION

For comparison purposes, two reference simulations were performed on a single processor using the two sets of valve duty cycles shown in Figures 4 & 5.

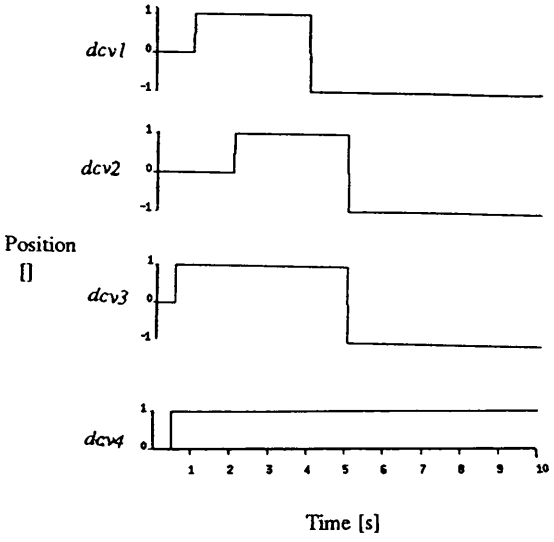


Figure 4 Control Valve Duty Cycles

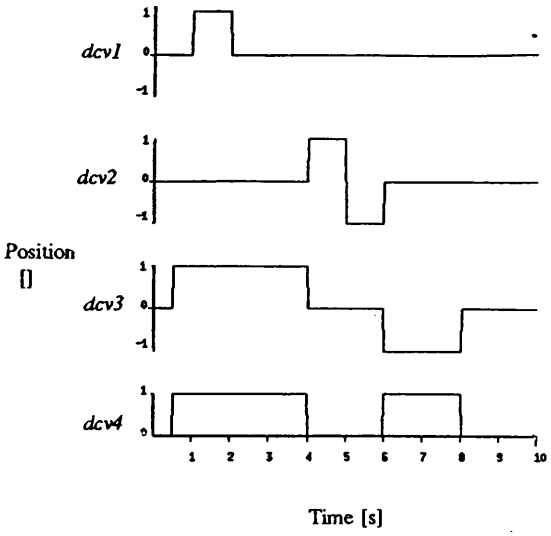


Figure 5 Alternate Valve Duty Cycles

The maximum pressure *error* permitted was 0.1 bar and a minimum allowable time step of 1μs was selected. Figures 6 & 7 show the variation in time step used by the **TLM** solver and the corresponding pressure *error*. At points of severe discontinuity (actuator travel limits, valve position changes etc.) successive time steps are rejected until the minimum allowable time step is reached, as shown in Figure-6. The simulated actuator positions and length of cable extended by the winch are shown in Figure 8 and the corresponding pressures at inlet and outlet in Figures 9, 10 & 11. The computational results presented here exhibit only minor differences from an equivalent circuit model developed using a lumped parameter **ODE** solver [Richards et al, 1990].

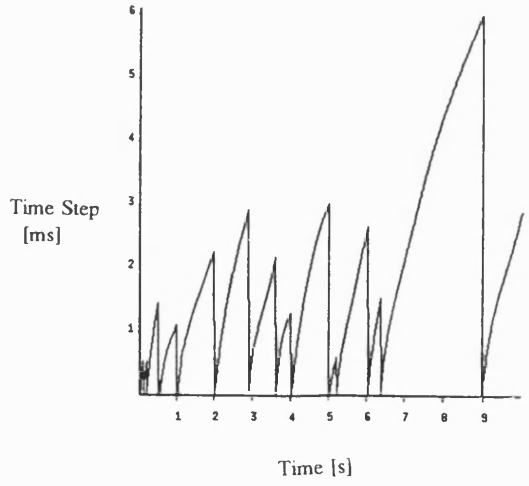


Figure 6 Time Step

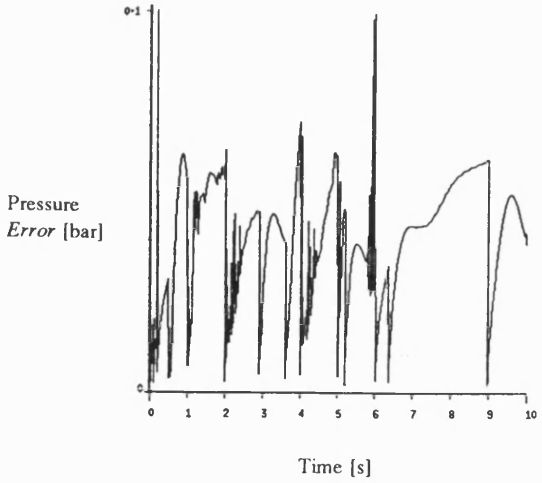


Figure 7 Pressure Error [bar]

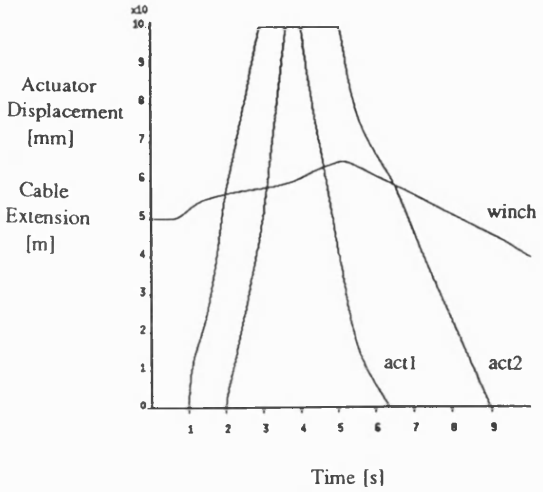


Figure 8 Actuator Positions & Cable Extension

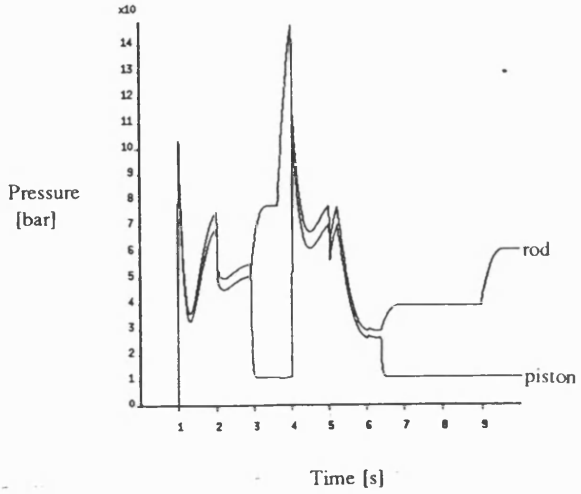


Figure 9 Actuator 1 Piston & Rod End Pressure

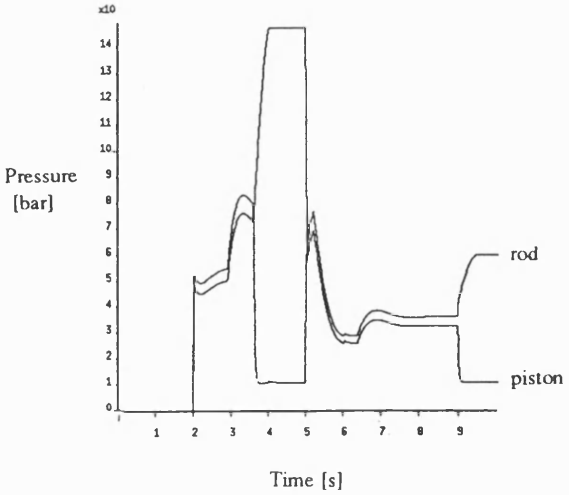


Figure 10 Actuator 2 Piston & Rod End Pressure

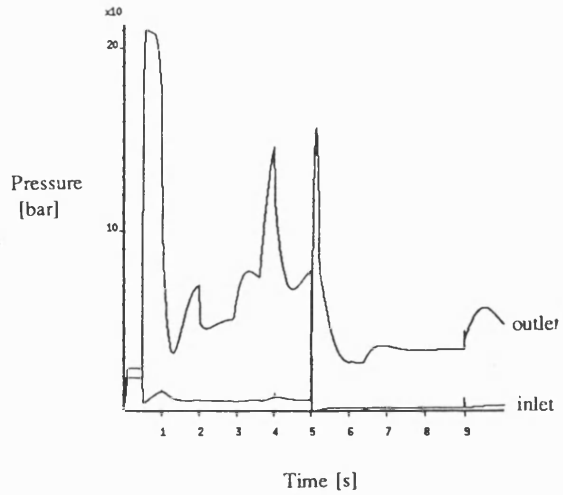


Figure 11 Motor Inlet & Outlet Pressures

MULTI-PROCESSOR SIMULATION

For the multi-processor study, the step rejection scheme was enabled and all sub-systems were synchronized, through the implementation of a central error controller, which sets the step size throughout the distributed simulation. The multi-processor simulation results are therefore identical to those obtained from the single processor computation.

The parallel computing hardware used consisted of an array of eight T800 transputers, including 2Mbytes available memory per processor (see Figure 12). To achieve the optimum increase in simulation speed *nearest neighbour* communications between processing elements is desirable. To achieve this the partitioned circuit must map *directly* onto the processor array. The software does permit the through-routing of messages via intermediate processors, but this is very much less efficient than communications over a single data link. The two, three and four processor configurations are shown in Figure 13. The master processor in each case is connected to the slave processors directly via the serial data links to ensure the fastest possible communications.

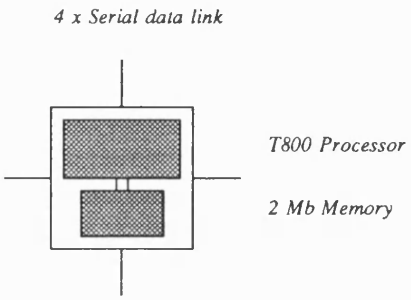


Figure 12 T800 TRAM

For the two processor simulation the actuator circuits were placed on one processor and the hydrostatic transmission on an adjacent processor. The step size controller was included in the latter partition. In the three processor parallel simulation the actuator circuits were separated and the step size controller placed onto the central actuator circuit to facilitate nearest neighbour data transfer between the respective partitions in a three processor chain. In the final four processor case studied the supply systems and directional control valves were placed onto the master processor, with the actuator and hydrostatic transmission circuits placed on separate slave processors.

Figure 14 shows the measured and *ideal* speed increase for the distributed simulations. [Note: the *ideal* speed increase for **n** processors is '**n**'). The results for the alternate valve duty cycles are also shown for comparison, which indicates a marginal improvement in speed up and distributed simulation efficiency. (Other valve duty cycles not shown differed only slightly from the results given). Figure 15 shows the consequent distributed simulation efficiency, defined as speed up divided by the number of processors (ideal speed up).

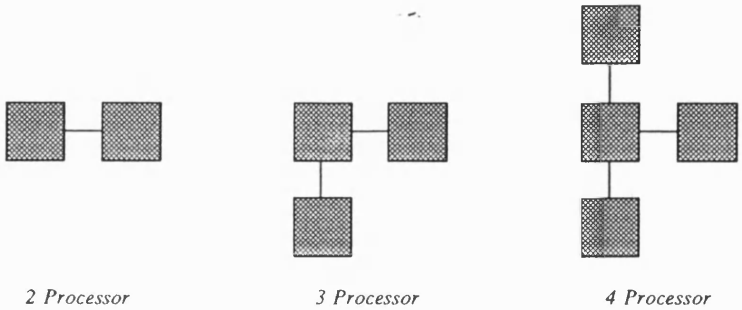


Figure 13 Two, Three & Four Processor Topologies

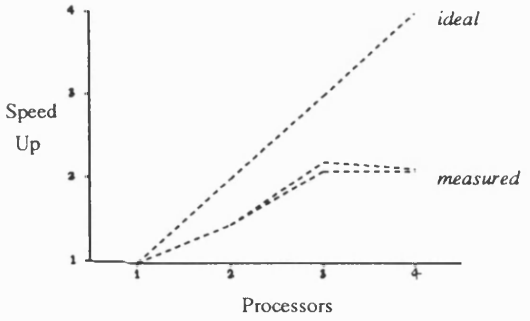


Figure 14 Measured & Ideal Speed Up

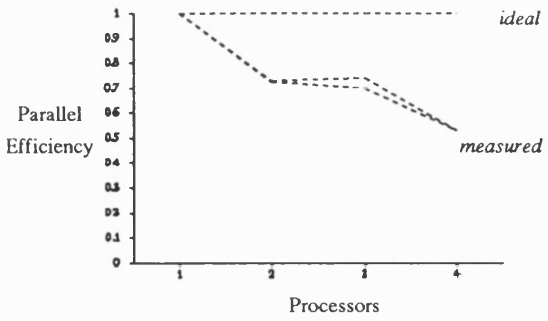


Figure 15 Measured & Ideal Parallel Simulation Efficiency

An indication of processor load may be obtained by measuring directly the time taken to compute all component models in each partition every simulation time step. Figures 16a, 16b & 16c show the computational load balance measured for each processor, for the two, three and four processor simulations respectively, using the valve duty cycles of Figure 4. The master process in each case also has a relatively small overhead associated with evaluating the accuracy of the current time step and calculating the next time step.

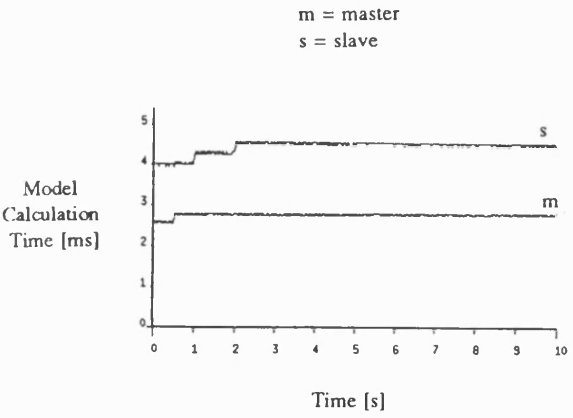


Figure 16a Two Processor Load Balance

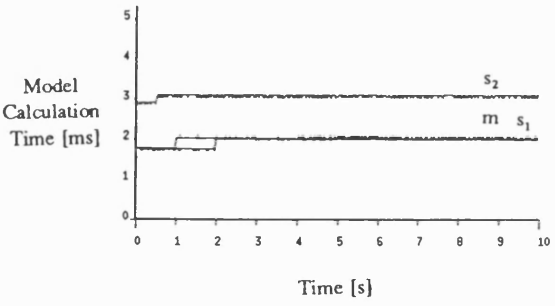


Figure 16b Three Processor Load Balance

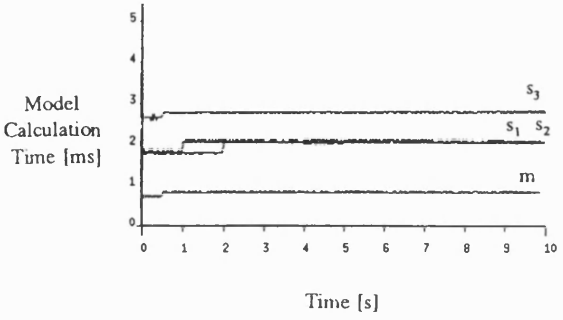


Figure 16c Four Processor Load Balance

DISCUSSION

The ideal speed up, equal to the number of processing elements, assumes zero time for communications between partitions and perfect balance of computational loads. In practice the data transfer between processors is a significant overhead and the relative computational efforts are difficult to determine when building the distributed simulation. Figure 14 clearly indicates the deviation of the measured from the ideal speed up for the example circuit. Figure 16a shows the different computational loads on both master and slave processors for the two processor simulation, measured in terms of the model calculation (computation) time per simulation time step. From Figure 15 the two processor parallel efficiency is only about 72 percent. This reduction in performance must be attributed to the unequal processor loads (approximately 60:40), in addition to the penalty associated with data transfer between processors.

The three processor simulation gave the best performance increase. In this case, the computational loads were reasonably balanced (see Figure 16b), yet the communications requirement was not excessive. Figure-14 indicates that the use of four processors is off-set by the increase in data transfer time and the imperfect computational load balance between the master partition (supply system and valves) and the slave partition assigned the hydrostatic transmission (denoted s_3 on Figure 16c). The re-partitioning to a four processor simulation scarcely influenced the computational load of the hydrostatic transmission sub-system from that of the three processor case (denoted s_2 on Figure 16b), in which case the consequent performance reduction was not surprising.

It is possible to partition this simulation further, but a distributed simulation in excess of five processors would require the use of *virtual* communications, because of the limitation to four serial data links per processor. Virtual communication necessitates processor *hops*, which is much less efficient than a direct data link. This configuration may be appropriate for even larger circuit simulations, but the increase in data transfer and the corresponding reduction in computational load per processor would reduce the performance. For satisfactory performance the ratio of model computation to data transfer must be large, although this is difficult to quantify using the currently available transputer utilities.

Computational load balancing is a very important factor when partitioning **TLM** based simulations, because of the nature of the simulation technique. **TLM** computations require all component models in the simulation to be executed before progressing to the next time step, thus no single partition may be more than one time step ahead of any other partition. Consequently, all partitions in the distributed simulation are 'locked-linked'; that is the computational speed is dependent upon the slowest partition. Fortunately, the data transfer requirement is typically small (only a few bytes), consisting of the characteristic pressures, line end pressures, step size and control information (step accept/reject). The data bandwidth is small, although the frequency may be quite high. Through the use of a software environment that generates bootable transputer object code, without the need for a transputer operating system, the communications penalty has been kept low.

The new generation of T9000 development transputers (not yet released) should enable much improved simulation speeds, as a result of increased processor power and data link speed, in addition to an anticipated maximum of six links per transputer. This hardware should facilitate a fast and efficient multi-processor simulations, incorporating up to seven processors in a master-slave configuration with direct communications.

CONCLUSIONS

A variable time step transmission line modelling (**TLM**) solver for the simulation of complex hydraulic systems has been partitioned onto multiple processors, facilitated by the propagation delay introduced by the pipelines. **TLM** enables hydraulic systems to be partitioned effectively onto master-slave type processor configurations, consistent with the implementation of a centralised error and step size controller. Ideally, the circuit topology should map directly onto the processor configuration, to avoid the use of virtual communications across one, or more intermediate processors. Direct communications, however, limits the master-slave configuration to a maximum total of five processors.

An example circuit was partitioned onto two, three and four processors. A maximum increase in performance (speed up) of 2.2 was recorded for the three processor configuration. An efficiency of some 72 percent was attributed to the overheads associated with communications between processors, as the processor loads were reasonably balanced. Extension to four processors had no effect on performance, as the increase in parallel operation was almost entirely off-set by increased data transfer and rather poor partitioning of two of the four sub-systems.

Changes in the duty cycle had little, or no effect on the performance of the distributed simulations, because each partition performed roughly the same number of instructions per time step, regardless of input conditions. If an iterative procedure had been incorporated into a model, however, this may have had a more significant effect.

ACKNOWLEDGEMENTS

This research is funded jointly by the UK Government, Science and Engineering Research Council, SERC Grant GR/F62971 and the Ministry of Defence. The authors gratefully acknowledge this support.

REFERENCES

Burton J D, Edge K A & Burrows C R, 1992, "Modelling Requirements for the Parallel Simulation of Hydraulic Systems", *Proc. ASME Winter Annual Meeting*, Anaheim, California, USA.

Hairer E & Wanner G, 1991, "*Solving Ordinary Differential Equations II, Stiff and Differential Algebraic Problems*". Springer-Verlag, Berlin Heidelberg.

Jansson A, Krus P & Palmberg J-O, 1992, "Variable Time Step Size Applied to Simulation of Fluid Power Systems Using Transmission Line Elements", *5th Bath International Fluid Power Workshop*, Bath University Fluid Power Centre, Bath, UK.

Krus P, Weddfelt K & Palmberg J-O, 1990, "Distributed Simulation of Hydro-Mechanical Systems", *3rd Bath International Fluid Power Workshop*, Bath University Fluid Power Centre, Bath, UK.

Pulko S H, Mallik A, Allen R & Johns P B, 1990, "Automatic Time-stepping in TLM Routines for the Modelling of Thermal Diffusion Processes", *International Journal of Numerical Modelling: Electronic Networks, Devices & Fields*, Vol 3, pp127-136

Richards C W, Tilley D G, Tomlinson S P & Burrows C R, 1990, "Type Insensitive Integration Codes for the Simulation of Fluid Power Systems", *Proc. ASME Winter Annual Meeting*, Texas, USA.

Sidell R S & Wormley D N, 1977, "An Efficient Simulation Method for Distributed-Lumped Fluid Networks", *ASME Journal of Dynamic Systems Measurement and Control*, pp34-40

Viersma T J, 1980, "*Analysis, Synthesis and Design of Hydraulic Servo-systems and Pipelines*", Elsevier SPC, Amsterdam, pp145-148.